

© Copyright 2009
Terry Hsin-Yi Shen

**Determining the Feasibility and Value of Federated Data
Integration with Combinations of Logical and Probabilistic
Inference for SNP Annotation**

Terry Hsin-Yi Shen

A dissertation
submitted in partial fulfillment of the
requirements for the degree of

Doctor of Philosophy

University of Washington
2009

Program Authorized to Offer Degree:
Medical Education and Biomedical Informatics

UMI Number: 3377322

Copyright 2009 by
Shen, Terry Hsin-Yi

INFORMATION TO USERS

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleed-through, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

UMI[®]

UMI Microform 3377322
Copyright 2009 by ProQuest LLC
All rights reserved. This microform edition is protected against
unauthorized copying under Title 17, United States Code.

ProQuest LLC
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106-1346

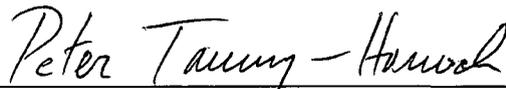
University of Washington
Graduate School

This is to certify that I have examined this copy of a doctoral dissertation by

Terry Hsin-Yi Shen

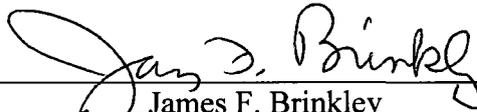
and have found that it is complete and satisfactory in all respects,
and that any and all revisions required by the final
examining committee have been made.

Chair of the Supervisory Committee:

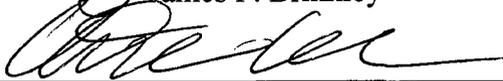


Peter Tarczy-Hornoch

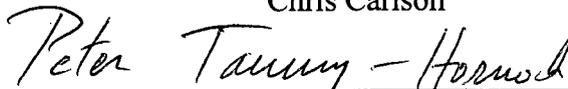
Reading Committee:



James F. Brinkley



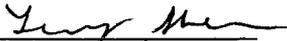
Chris Carlson



Peter Tarczy-Hornoch

July 20, 2009

In presenting this dissertation in partial fulfillment of the requirements for the doctoral degree at the University of Washington, I agree that the Library shall make its copies freely available for inspection. I further agree that extensive copying of the dissertation is allowable only for scholarly purposes, consistent with "fair use" as prescribed in the U.S. Copyright Law. Requests for copying or reproduction of this dissertation may be referred to ProQuest Information and Learning, 300 North Zeeb Road, Ann Arbor, MI 48106-1346, 1-800-521-0600, to whom the author has granted "the right to reproduce and sell (a) copies of the manuscript in microform and/or (b) printed copies of the manuscript made from microform."

Signature 

Date 7/24/05

University of Washington

Abstract

Determining the Feasibility and Value of Federated Data Integration with
Combinations of Logical and Probabilistic Inference for SNP Annotation

Terry Hsin-Yi Shen

Chair of the Supervisory Committee:
Professor Peter Tarczy-Hornoch
Department of Medical Education and Biomedical Health Informatics

Most common and complex diseases are influenced at some level by variation in the genome. The future work of statistical geneticists, molecular biologists, and physician-scientists with interests in genetics or genomics must thus take genetics into consideration. Research done in public health genetics, specifically in the area of single nucleotide polymorphisms (SNPs), is the first step to understanding human genetic variation. Functional uncertainty regarding SNP function, volume of information, and cost of investigating potential causative SNPs result in the prioritization of SNPs to be an important driving problem for this dissertation. SNP Integration Tool (SNPit) is a data integration system that looks at all the possible

predictors of functional SNPs and provides the user with integrated information and decision making capability. Determining the feasibility and value of a data integration system combinations of probabilistic and logical inference for functional SNP annotation is the main focus of this dissertation. The dissertation describes challenges from both the biological and biomedical informatics standpoint regarding how to represent, integrate, and conduct inference over disparate biological data sources. Through development of these systems, this dissertation studies the feasibility of this form of federated data integration for SNP annotation and also assesses its' accuracy for SNP annotation. The work characterizes the suitability of combinations of logical and probabilistic inference with federated data integration for both point and regional SNP annotation. This dissertation contributes to our knowledge in the area of data integration and biological applications by describing the design, implementation, and evaluation of combinations of logical, probabilistic, and both logical and probabilistic inference applied to the domain of functional SNP annotation.

Table of Contents

	Page
List of Figures.....	vi
List of Tables.....	ix
Glossary	x
Chapter 1: Introduction.....	1
1.1 Overview	1
1.2 Biological Background and Significance	2
1.2.1 Use of SNPs in Genome Wide Association Studies.....	3
1.2.2 Difficulty Predicting Functional SNP Annotation	3
1.2.3 Improvements Needed for Functional SNP Annotation	4
1.2.4 Impact of GWAS Studies on potential for Predictive, Preventive, Personalized Medicine	5
1.3 Research Questions and Aims	7
1.3.1 Characterize feasibility of federated data integration for SNP annotation and assess its' accuracy	7
1.3.2 Characterize feasibility of combining federated data integration with logical inference or probabilistic inference for point and regional SNP annotation and assess its' accuracy	8
1.3.3 Characterize feasibility of federated data integration combining both logical and probabilistic inference for point and regional SNP annotation and assess its' accuracy and utility	9
1.4 Related Work on SNP annotation and on data integration.....	10
1.5 Research Methods	11
1.6 Findings and Contributions of this Dissertation.....	12
1.7 Outline of Dissertation	15
Chapter 2: Background on Genome Wide Association Studies and SNPs	17

2.1	Introduction	17
2.2	Genome Wide Association Studies and SNPs.....	19
2.2.1	Single Nucleotide Polymorphisms (SNPs) as Markers	21
2.2.2	Definition of Genome Wide Association studies	23
2.2.3	Genotyping and statistical analysis tools.....	26
2.2.4	Why are SNPs used in genome wide association studies.....	28
2.3	Challenges to applying results of GWA studies.....	30
2.3.1	Contrasting clinical and statistical significance	31
2.3.2	Law and Ethics	39
2.3.3	Application of GWA studies	40
2.3.3.1	Pharmacogenetics.....	41
2.3.3.2	Direct to Consumer Genetic Tests.....	43
2.3.3.3	Public Health Surveillance	45
2.3.3.4	Elucidation of pathogenic pathways.....	46
2.3.4	Stakeholders in understanding role of genetic variation	47
2.4	Discussion.....	48
Chapter 3: Current Approaches and Challenges to Functional Annotation of SNPs...		
.....		50
3.1	Introduction	50
3.2	Types of Functional SNPs	51
3.2.1	SNPs Involved with Transcription	51
3.2.2	SNPs Involved with Translation.....	54
3.2.3	SNPs Involved with Gene Regulation.....	54
3.2.4	Challenge to capturing information on alternative slicing	55
3.3	Manual Approaches to SNP Annotation	56
3.4	Automated and semi-automated approaches to SNP Annotation.....	57
3.4.1	Review of current approaches to SNP annotation.....	58
3.4.2	Limitations of Current Approaches	59
3.5	Challenges to evaluation of SNP annotation systems/approaches	61
3.6	Discussion.....	63

Chapter 4: Foundations of the SNPit Data Integration System.....	65
4.1 Introduction	65
4.2 Data Integration Concepts	65
4.2.1 Types of Data Integration- How the data is stored.....	66
4.2.2 Types of Data Models – How the data is represented.....	67
4.2.3 Types of User Interfaces – How the data is displayed	68
4.3 BioMediator Data Integration System.....	69
4.3.1 Overview	70
4.3.2 Architecture	70
4.3.3 Mediated Schema	71
4.3.4 User Interfaces.....	72
4.4 Data Integration and Logical Inference.....	73
4.4.1 Overview	74
4.4.2 Jess Logical Inference Engine.....	74
4.4.3 Prior Work Into Logical Inference and BioMediator	75
4.5 Data Integration and Probabilistic Inference.....	76
4.5.1 Overview	76
4.5.2 Uncertainty: The Uncertainty In Information Integration (UII) Project	77
4.5.3 Prior Work Into Probabilistic Inference and BioMediator	81
4.6 Discussion.....	81
 Chapter 5: Determining Feasibility via Implementation of the SNPit System	 83
5.1 Introduction	83
5.2 Design of the SNPit System	83
5.2.1 Building upon BioMediator Core System	83
5.2.2 Building upon Prior Logical Inference Extensions to BioMediator.....	85
5.2.3 Building upon Prior Probabilistic Inference Extensions to BioMediator	86
5.3 Implementation of the SNPit System	86
5.3.1 Data Sources.....	87

5.3.2	Common Data Model for SNPit.....	92
5.3.3	Logical Inference for SNPit.....	96
5.3.4	Probabilistic Inference for SNPit.....	100
5.3.5	User Interfaces.....	104
5.4	Evolution of the SNPit System.....	106
5.4.1	SNPit v1 – Data integration alone, no ranking.....	107
5.4.2	SNPit v2 – SNPit v2 – Ranked SNPs using logical inference	109
5.4.3	SNPit v3 – Ranked SNPs using probabilistic inference.....	112
5.4.4	SNPit v4 – Ranked SNPs using both logical and probabilistic inference	115
5.5	Discussion	116
Chapter 6: Assessing the Value of the SNPit System		119
6.1	Introduction	119
6.2	Approach to gold standard and metrics.....	119
6.2.1	Challenge to finding a gold standard.....	120
6.2.2	Challenge to evaluating clinical significance	122
6.2.3	What metrics were chosen and why	123
6.2.3.1	Sensitivity/Specificity	123
6.2.3.2	Precision/Recall.....	125
6.2.3.3	Accuracy.....	127
6.3	Calibration Analysis of data sources	128
6.3.1	Calibration analysis for nonsynonymous/synonymous, splicing, and transcription factor categories	128
6.4	Evaluating the different versions of SNPit.....	133
6.4.1	Evaluating SNPit v1 – Data integration alone.....	134
6.4.1.1	Categories used in this evaluation.....	134
6.4.1.2	Category by category evaluation (Synonymous/Nonsynonymous, Splicing, and Transcription Factor Binding Separately), using sensitivity and specificity	135

6.4.1.3 Aggregate cross category evaluation (Combining Synonymous/Nonsynonymous, Splicing, and Transcription Factor Binding), using sensitivity and specificity	138
6.4.2 Evaluating SNPit v2 – Ranked SNPs using logical inference, using recall and precision measures.....	140
6.4.3 Evaluating SNPit v3 – Ranked SNPs using probabilistic inference, using recall and precision measures.....	144
6.4.4 Evaluating SNPit v4 – Ranked SNPs using both logical and probabilistic, using recall and precision measures	146
6.4.5 Cross cutting evaluation of SNPit v1-v4	151
6.5 Measuring the Usability and Usefulness of SNPit, User based evaluation.....	156
6.6 Discussion.....	159
Chapter 7: Conclusions.....	119
7.1 Introduction	161
7.2 Contributions.....	161
7.3 Limitations.....	163
7.4 Future Work.....	165
7.5 Concluding Remarks	167
Bibliography	169
Appendix A: Wrappers.....	182
Appendix B: Protégé Data Model	296
Appendix C: Belief Resolver java files	360
Appendix D: Jess rules implementing decision tree.....	381
Appendix E: Qualitative study consent form and questions	386

List of Figures

Figure 2.1. Diagram of single nucleotide polymorphism.....	21
Figure 2.2. Diagram of common SNPs in a population of individuals and SNPs that are associated to the phenotype	24
Figure 2.3. Epidemiologic 2x2 or 2x3 table for case control studies	25
Figure 2.4. Illumina and Affymetrix SNP microarray platforms	28
Figure 2.5. Genome wide association study results for rheumatoid arthritis	33
Figure 2.6. ROC curves for 3 Type 2 diabetes predictive tests.....	36
Figure 3.1. General diagram of the process of transcription and translation	53
Figure 3.2. The mechanics of alternative splicing, silencers and enhancers	55
Figure 3.3. Recent survey findings that coding SNPs have lower minor allele frequencies.....	62
Figure 4.1. Overview diagram of BioMediator system.....	71
Figure 4.2. Graphical based interface implemented using the Touchgraph.....	73
Figure 4.3. Basic connection between a Query (Q) and a Result (R)	79
Figure 4.4. Extended network showing nodes(N) and edges(e).....	80
Figure 5.1. Diagram of the SNPit system.....	85
Figure 5.2. Mediated schema of SNPit.....	93
Figure 5.3. The common data model is implemented via the Protégé Knowledge Base	95
Figure 5.4. Literature summary of the various methods in which the location on which a SNP resides impacts the potential function of the SNP	97
Figure 5.5. Decision tree with heuristic weights assigned to the branches	99
Figure 5.6. SNPit Touchgraph graphical user interface	105
Figure 5.7. Overall design of the SNPit website	106
Figure 5.8. Input screen for main search option.....	108
Figure 5.9. Returned results for SNP rs405509 using the Main Search option.....	109
Figure 5.10. Jess rule for nonsynonymous SNPs	110
Figure 5.11. Input screen for logical rules ranking page.....	111
Figure 5.12. SNPit display of ranked SNPs based on the logical rules implemented through Jess	111

Figure 5.13. Central SNP presented as a flower analogy, with additional attributes acting as petals extending out from the central seed	113
Figure 5.14. Probabilistic results for SNPit.....	114
Figure 5.15. SNPit interface demonstrating the ranked SNPs that are returned when users select the Probabilistic Inference for Regional SNPs option	115
Figure 5.16. SNPit interface demonstrating the ranked SNPs that are returned when users select the Probabilistic and Logical Inference for Regional SNPs option	116
Figure 6.1. Recall and precision measured using HGMD and dbSNP.....	126
Figure 6.2. Graph of the probability of a random SNP from dbSNP being in HGMD given some score for the SIFT category	131
Figure 6.3. Graph of the probability of a random SNP from dbSNP being in HGMD given some score for the BDGP category	132
Figure 6.4. Graph of the probability of a random SNP from dbSNP being in HGMD given some score for the TFSearch category	133
Figure 6.5. Figure demonstrates the final results of all the sensitivities and specificities of the candidate genes, pertains to evaluation of SNPit v1	137
Figure 6.6. Both cycles of the first phase of evaluation for the missense/nonsense, splicing and regulatory categories	139
Figure 6.7. ROC curve for SNPit v2 – logical inference, indicating good predictive power because the graph line curves along the upper left corner.....	141
Figure 6.8. Recall/precision curve for SNPit v2 – logical inference, indicating good predictive power because the graph line curves along the upper right corner	142
Figure 6.9. Classification of the SNP functional groups for logical inference rules separated by SNPs randomly run through HGMD and dbSNP.....	143
Figure 6.10. ROC curve for SNPit v3 – probabilistic inference	145
Figure 6.11. Breakdown of the different categories that provided information to the SNP score, which contributed to the probabilistic inference algorithm.....	146
Figure 6.12. Various approaches to combining logical and probabilistic scores	148
Figure 6.13. ROC curves of the five different methods used to combine the logical and probabilistic scores	150
Figure 6.14. ROC curve for SNPit v4 using the method that performed the best, the customized probabilistic score was multiplied with the logical score.....	151
Figure 6.15. ROC curve for multiple comparison between SNPit results ranked by probabilistic inference, results ranked by logical inference, results ranked by probabilistic inference, and results ranked randomly with no inference.....	153

Figure 6.16. Precision/recall curve for multiple comparison between SNPit results ranked by probabilistic inference, results ranked by logical inference, results ranked by probabilistic inference, and results ranked randomly with no inference..... 154

List of Tables

Table 2.1. Selected list of recent genome wide association studies	20
Table 2.2. List of genes that influence adverse drug effects	42
Table 2.3. List of different genetic tests currently available to the public	44
Table 3.1. Tables includes the comparisons of different previous SNP annotation systems	58
Table 4.1. Uncertainty metrics for four different measures	78
Table 5.1. List of data sources that SNPit links to, along with brief descriptions of each source	88
Table 5.2. Database and entities for the common data model.....	94
Table 5.3. Descriptions of Ps and Pr values and the rationale for who the values were selected	100
Table 5.4. Descriptions of Qs and Qr values and the rationale for who the values were selected	102
Table 6.1. Table of TP, FN, FP, TN for classifier systems performances	127
Table 6.2. Various formulas that can be created by using the labels in the table of True Negatives, False Positives, False Negatives, and True Positives	128
Table 6.3. Spreadsheet of how we arrive at the calculation of the probability of B A, which equates to the probability of a SNP being found in HGMD given a change in its score	129
Table 6.4. Splicing category evaluation for five candidate genes, pertains to version 1 of SNPit	137
Table 6.5. The following table demonstrates how the sensitivities and specificities for the across category evaluations compare to one another	139
Table 6.6. Portion of the spreadsheet used to create the ROC curve for SNPit v4 with both probabilistic and logical inference	148
Table 6.7. Accuracy measures for ranked list and no ranked list versions of SNPit	155

Glossary

ALLELES: Alternate forms of a gene.

ANALYTIC VALIDITY: The ability of a genetic test to accurately and reliably predict the genotype of interest.

CLINICAL VALIDITY: The ability of a genetic test to predict the final phenotype trait.

CLINICAL UTILITY: The ability of a genetic test to provide useful health outcomes as compared to traditional health treatment without genetic tests.

COMMON SNPS: SNPs whose minor allele frequency are greater than 5%.

DATA INTEGRATION: The process of querying across heterogeneous data sources and combining the results.

FUNCTIONAL SNP ANNOTATION: Determining and documenting the biological outcome of a SNP.

GENE: A portion of DNA that eventually codes for a protein.

GENOME WIDE ASSOCIATION STUDY (GWAS): A study of the association between a variant and an outcome by using markers across the genome.

HAPLOTYPE: Alleles along the same chromosome.

LOCUS: A genetic location on a chromosome.

LINKAGE DISEQUILIBRIUM: When alleles are inherited in blocks due to their proximity to one another.

LOGICAL INFERENCE: A form of inference where conclusions are made after certain premises are presented.

POINT SNP: The individual SNP itself.

PROBABILISTIC INFERENCE: Inference conducted using probabilistic metrics.

REGIONAL SNP: The individual SNP and all the SNPs in linkage disequilibrium with that SNP.

SINGLE NUCLEOTIDE POLYMORPHISMS (SNP): A change in one nucleotide base pair along the genome.

SNP CHIP: A type of microarray that can identify SNPs in an individual.

TAG SNPS: Single nucleotide polymorphisms that due to linkage disequilibrium are highly correlated with a large number of other SNPs and thus, act as a proxy in genome wide association studies.

Acknowledgements

I would like to acknowledge the following groups and individuals for helping me throughout my doctoral degree journey:

- To Peter Tarczy-Hornoch, for being the best advisor any student could ask for, thank you for your guidance and mentorship through the years.
- To Chris Carlson, for your colloration and mentorship and for giving me the idea to pursue SNPs in the first place.
- To the Biomedical Data Integration and Analysis (Bio-DIAG) team for their feedback and help throughout the implementaion of the SNPit system: Todd Detwiler, Ron Shaker, Wolfgang Gatterbauer, Eithon Cadag, Brent Louie, and Janos Barbero.
- To the National Library of Medine and the National Science Foundation for providing me with the funds to be able to pursue my doctoral degree and research.
- Finally, to my parents, Sam and Doris Shen, for bringing me to the United States when I was five and providing me with the educational opportunitis that they themselves never had.

Chapter 1: Introduction

1.1 Overview

We are now entering a revolutionary stage in the history of science and medicine. Due to advances in both computing and genotyping capabilities, we can potentially take a proactive approach to the diagnosis and eventual treatment of a disease (1-3). Many diseases, both common and rare, have a genetic component, and uncovering the etiology of a disease can be facilitated by understanding this genetic basis (4). The vision espoused by many, including Hood and Zerhouni (2, 3), of personalized, predictive, preventive medicine tailored to the individual is at its foundation based on an assessment of genetic risk.

Key to this vision is understanding the association between an individual's genetic makeup (genotype) and risk for disease or response to therapy (elements of phenotype). Uncovering the potential genetic contribution to a disease can be done through a variety of techniques, with genome wide association studies being one of the newer and more promising approaches. Genome wide association studies (GWAS) are defined by the National Institute of Health as a study of genetic variants across the human genome with the purpose of finding associations between the markers being studied and observable traits (5). A key step in translating from statistical association to causality, in order to intervene via prevention or treatment, is understanding the functional impact of the genetic variability. One of the most

common markers of genetic variability used in association studies today are single nucleotide polymorphisms (SNPs). The process of functional annotation of genetic variability today (SNPs in particular) is largely manual. The overarching aim of this dissertation was to determine the feasibility and value of federated data integration with combinations of logical and probabilistic inference for SNP annotation. In the following sections an overview of the structure, content and findings of the dissertation is presented.

1.2 Biological Background and Significance

Traditionally genetic variability was assessed indirectly using linkage studies with relatively sparse markers or through sequencing of individual genes. Today researchers are using genome wide approaches to link variability in phenotype to underlying genetic changes. These approaches are based on the fact that variability in regulatory and coding regions of DNA impact on protein transcription and regulation, which in turn can have biologic or clinical impact.

DNA is a key building block of life, it is composed of four nucleotide base pairs (adenine, thymine, guanine, or cytosine) stranded in double helix. A portion of DNA, a gene, contains the information needed to make a protein via the processes of transcription and translation (6). The human genome is comprised of around 3.3 billion base pairs, 99% of which is identical between any two individuals, variations

within the population that occur at a frequency greater than 1% is called a polymorphism (7). Details on the biological background of polymorphisms are presented in section 2.2 of Chapter 2.

1.2.1 Use of SNPs in Genome Wide Association Studies

The ongoing search for disease susceptible loci have led to the use of genome wide association studies (GWAS), which measures the correlation between many alleles and a disease in the population. Single nucleotide polymorphisms (SNPs) are the most common form of human variation and due to their frequency of occurrence as well as ease of genotyping, they are the pre-dominant form of markers used in genome wide association studies. Additional details and benefits of genome wide association studies are presented in section 2.2.1 of Chapter 2. Once a correlation is found in a GWAS, the next step is to understand the biologically based causal relationship of the result, which might guide diagnosis, intervention, and treatment. Functional annotation is the process of trying to predict the biological outcome of a SNP.

1.2.2 Difficulty of Predicting Functional SNP Annotation

The prediction of the functional impact of a SNP (functional annotation) is challenging for a number of reasons. The location where a SNP resides along a

genome can give us some preliminary insight into the ultimate function of that SNP; for example, SNPs that reside in introns are spliced out the gene (thus non-coding) and thus thought to have low predictive power in terms of the final protein product and thus on ultimate biological function and phenotypic impact as compared to exonic SNPs. In addition genes interact with other genes and proteins as well as environmental exposures, which further complicates the ultimate prediction of functional impact of a given SNP and related phenotype. Chapter 3 of this dissertation details the various molecular mechanisms that are involved with predicting the functional impact of a SNP.

1.2.3 Improvements Needed for Functional SNP Annotation

Non-reproducible genome wide association studies have become such a concern to the research community (8) that studies are now required to include reproduced results in order to be published. The cause for these spurious results can be due to a variety of reasons including: the design of the study was not valid or due to statistical flaws. To increase the chances of finding true causal association in a GWAS, researchers need to conduct a valid GWAS study as well as provide a biological reasoning for why a SNP is associated with a disease. To improve the ability for researchers to uncover the functional annotation of a SNP, we need to collection, integrate, and analyze the information that is available on SNPs in public databases.

Current approaches to SNP annotation require manually retrieving information on SNP annotation one at a time and then integrating the sources together by hand.

Automated annotation saves both time and resources by using computers to systematically collect the information and merge them into one repository. Chapter 3 of this dissertation discusses the difficulties in annotating functional SNPs.

This dissertation aims to develop and implement methods, models and tools to facilitate functional annotation (Chapter 5) and evaluates these approaches (Chapter 6).

1.2.4 Impact of GWAS Studies on Predictive, Preventive, Personalized Medicine

The importance of GWAS and functional SNP annotation is that predictive, preventive, and personalized medicine is potentially where the future of healthcare and public health lies. Rather than the current model of treating diseases once the disease has been diagnosed, advances in both genotyping and molecular diagnostics via blood samples have opened up the potential of predicting the future health of individuals (2). Personalized medicine resulting from genetic variation studies can occur when discoveries from genetic variation analyses lead to new drug targets and/or the improvement of existing drugs (9) tailored to the individual. The discoveries from genetic variation studies can also be used to identify subsets of

patients for surveillance purposes permitting targeting of other screening methods to higher risk populations. Using molecular understanding to reclassify and identify pre-symptomatic disease is a vision articulated by both Hood and Zerhouni as being a core challenge and opportunity for science in this century (2, 3).

These predictive and preventative breakthroughs, if they live up to their promise, will surely change the way healthcare is delivered to the general public. Predictive, preventive, and personalized (P3) medicine is a vision that will directly address the health goal of Healthy People in Every Stage of Life. Though today a full genome SNP analysis is about \$1000, the cost continues to drop rapidly and unlike other tests it needs to be done only once for a patient and it applies not just to one disease but to all diseases with a strong genetic component. Already the cost for a full genome scan is in the range of other very widely used tests such as CTs, MRIs, and ultrasounds.

Looking beyond the care of individual patients, the mission of public health is to promote the population's health and prevent diseases. As public health practitioners often refer to the leading causes of mortality to determine where to prioritize resources and funding, it is important to highlight that nine of the top ten leading causes of mortality in the US include genetic components (10). The human population shares 99% of the same DNA, but that 1% difference is an important factor as to why certain people get a disease while others do not, and why some respond to therapy better than others (11). To really reduce the burden of disease at

a population level, we must understand the genetic variation that occurs in individuals. This will permit targeting of other screening tests and interventions across the population to those who most need and benefit from them.

Section 2.3 of Chapter 2 details the various ways that the results from genome wide association studies can be applied in a clinical setting and thus, how it can help bridge the gap between biological discoveries and predictive, preventive, and personalized medicine.

1.3 Research Questions

As stated in the introduction, the main question of this dissertation is if it is both feasible and valuable to create and evaluate a functional SNP annotation system using federated data integration as the foundation and building logical, probabilistic and probabilistic with logical inference over the returned data.

1.3.1 “Is it feasible to create a federated data integration for SNP annotation and how accurate is such a system?”

The first component to answering the overall question of this dissertation is to build the basic federated data integration system for the purpose of functional SNP annotation. Chapter 5, sections 5.2 and 5.3, describes the design and implementation of the SNPit (SNP Integration Tool) system. Accuracy for the federated integration

component of SNPit is measured using sensitivity and specificity and is done both within and across categories, and our results have demonstrated that combining the information across categories does indeed improve sensitivity ; chapter 6, section 6.4 discusses the evaluation of the federated integration component of SNPit. Chapter 5, sections 5.2 and 5.3, also describes adaptations needed for data integration to work in the context of functional SNP annotation.

1.3.2 “Characterize feasibility of combining federated data integration with logical inference or probabilistic inference for point and regional SNP annotation and assess its’ accuracy?”

The second component to answering the overall question of this dissertation is to independently include probabilistic inference and logical inference for both point and regional SNPs (the individual SNP and those in linkage disequilibrium with that SNP) on top of the base federated data integration and assess its accuracy. Chapter 5, section 5.3.3, describes the design and implementation of the probabilistic inference component of SNPit. Chapter 5, section 5.3.4, describes the design and implementation of the logical inference component of SNPit. We created a custom SNP algorithm that looked at the probabilistic scores of the transcription factor binding, splice sites, protein function prediction, and evolutionary conservation of a given SNP, and created a central probabilistic score for the SNP, then applied the

uncertainty information integration model algorithm to the results. This demonstrated the limitations to generalizability of prior approaches to probabilistic data integration. Chapter 6, section 6.4.3 describes the recall and precision evaluation of this question.

1.3.3 ““Is it feasible to create a federated data integration combining both probabilistic and logical inference for point and regional SNP annotation and how accurate and useful is such a system?”

The third component to answering the overall question of this dissertation is to combine both probabilistic and logical inference for both point and regional SNPs on top of the federated data integration and assess its accuracy. The rationale for developing and evaluating this combined approach is that the two inferential approaches utilize different kinds of data for ranking data results. The hypothesis was that the combination would be more powerful than either alone. Chapter 5, section 5.4.4, describes the design and implementation of the combined probabilistic and logical inference component of SNPit. For this version, we created a custom decision tree that looked at location of the SNP (coding vs. non-coding, UTR vs. non-UTR, etc) and assigned heuristic weights to each category. The heuristic weights were then transformed into a probability and used to modify the probability score for the SNP. We then applied the uncertainty information integration model

algorithm to the results. Chapter 6, section 6.4.4 describes the recall and precision evaluation of the combined approach

1.4 Related Work on SNP annotation and on data integration

Chapter 3, section 3.3., describes the manual approach to annotation which has as key limitations as compared to automated and semi-automated approaches an increase in the amount of time and effort necessary to conduct annotations as well as the element of human error. Chapter 2, section 3.4 details the previous automated/semi-automated systems that have dealt with SNP annotation, table 3.1 describes the main components of each of the previous SNP annotation systems. The key limitations of other systems/ are that: none used a federated data integration approach when it comes to integration of data sources, none of them use probabilistic and logical inference for the purpose of functional SNP prediction, and none conducted formalized evaluation of their systems. Our system, SNPit, is unique in that it is the first SNP annotation system that uses a federated data integration approach as well as both probabilistic and logical inference to predict SNP functionality. Our system is also the first to conduct a formal evaluation of our annotation techniques.

Chapter 4 presents the foundations on which the SNPit system was built. Section 4.3 describes the BioMediator federated data integration system, which SNPit builds on top of to retrieve information from different data sources and

integrate together. Section 4.4 describes the concept of probabilistic inference and previous work building off of probabilistic inference and BioMediator. Section 4.5 describes the concept of logical inference and previous work building off of logical inference and BioMediator.

1.5 Research Methods

The foundations of the SNPit data integration system are discussed in section 5.2 of Chapter 5. The key elements are the data source, the common data model, the probabilistic inference component, the logical inference component, and the user interfaces. SNPit builds on top of BioMediator, which is composed of wrappers, a metawrapper, a source knowledge base, and user interfaces (section 4.3). Logical inference was added in the form of expert knowledge rules in the form of a decision tree (section 4.4.). Probabilistic inference was added incorporating uncertainty metrics and network reliability theory (section 4.5.).

Chapter 5 details the various versions of the SNPit system. Section 5.3 presents the different elements of SNPit, including its data sources, common data model, logical inference extensions, probabilistic inference extensions, and user interfaces. Section 5.4 details the four versions of SNPit developed for this dissertation.

The assessment of the value of SNPit using a combination of sensitivity / specificity, recall / precision, and accuracy are discussed in Chapter 6. Section 6.4.1

examines the evaluation of SNPit v1, consisting of the federated data integration system. Section 6.4.2 examines the evaluation of SNPit v2, consisting of ranked SNPs using logical inference. Section 6.4.3 examines the evaluation of SNPit v3, consisting of ranked SNPs using probabilistic inference. Section 6.4.4 examines the evaluation of SNPit v4, consisting of ranked SNPs using both logical and probabilistic inference.

1.6 Findings and Contributions of this Dissertation

The overarching aim of this dissertation was to determine the feasibility and value of federated data integration combining logical and probabilistic inference for SNP annotation. The key findings related to determining the feasibility of a federated data integration combining logical and probabilistic inference for SNP annotation were:

- The baseline SNPit system with federated data integration was indeed feasible using the general BioMediator framework. SNP annotation can be modeled in a mediated schema, wrappers can be created for SNP data sources.
- SNPit with logical inference can be created using expert knowledge and existing biological principles. Codified logical inference can accurately capture human heuristics related to SNP annotation.

- SNPit with probabilistic inference is indeed feasible by summarizing the various uncertainties that exist within and between different sources and records.
- The general Uncertainty in Information Integration framework and network reliability approach did not generalize to functional SNP annotation. The topology of the graph required a new approach to combining probabilities.
- It is indeed feasible to combine logical and probabilistic inference for the purposes of predicting the functional annotation of a SNP. A number of different approaches for combining the probabilistic and logical ranking/pruning of the result graph needed to be evaluating

The key findings related to determining the value of a federated data integration combining logical and probabilistic inference for SNP annotation were:

- SNPit baseline version with federated data integration improves the sensitivity of returned results as compared to a system without federated data integration.
- SNPit with logical inference has very good predictive power when it comes to functional SNP annotation.
- SNPit with probabilistic inference has only moderately good predictive power with it comes to functional SNP annotation.

- SNPit combining logical and probabilistic inference has very good predictive power when it comes to functional SNP annotation. Despite the fact the two forms of inference rely on different types of data, combining the two forms of inference does not improve performance over logical inference alone.
- User evaluations of the SNPit system have demonstrated positive results with users feeling that the system is both useful and usable.

The contributions of this dissertation are:

- The design and implementation of a new semi-automated SNP annotation system that uses a general purpose federated data integration system (BioMediator) with a custom mediated schema and wrappers (Chapter 5, section 5.4) demonstrating both the feasibility and value (though limited) of such a system without pruning the result graph for SNP annotation contributing to both our biomedical informatics and biological knowledge base.
- The creation of a system allowing probabilistic and/or logical inference over an integrated data set (Chapter 5, section 5.4) demonstrating feasibility of combining logical and probabilistic inference over integrated biological data which heretofore had not

been done contributing to both the data integration and biomedical informatics knowledge base.

- The finding that the approach to federated data integration and logical inference previously used was generalizable in contrast to the approach used for probabilistic inference which did not generalize.
- The demonstration of the value of adding inference to data integration for purposes of functional SNP annotation.
- The paradoxically relatively small marginal added value of combining the two vs. logical inference alone contributing to the data integration, biomedical informatics, and biological knowledge bases
- An evaluation of our SNPit system in which we demonstrate that the use of federated data integration with both probabilistic and logical inference provide strong prediction for functional SNP annotation contributing to the biomedical informatics and biological knowledge base.
- The incorporation of users in evaluations was worthwhile, SNPit was deemed to be both usable and useful for our sample of users.

1.7 Outline of Dissertation

There is currently a vision of predictive, preventive, and personalized medicine that researchers are striving towards. To achieve this vision, the genetic basis of disease needs to be understood. Genome wide association studies and SNPs play a significant role in understanding the genetic mechanisms of disease. Chapter 2 discusses the background on genome wide association studies and SNPs. The current approaches and systems available for SNP annotation are limited and not scalable. Chapter 3 is an overview of the current approaches and challenges when it comes to SNP annotation. Chapter 4 summarizes the base upon which the SNPit system was built. Chapter 5 deals with determining the feasibility and implementation of the SNPit system, where use federated data integration with probabilistic and logical inference for the purposes of SNP annotation. Finally, Chapter 6 deals with the evaluation of the various versions of SNPit. This dissertation ends in Chapter 7 with a summary of the lessons learned, limitations of the system, limitations of the evaluation of the system, and future directions of the work.

Chapter 2: Background on Genome Wide Association Studies and SNPS

2.1 Introduction

Molecular genetic studies extend back to the 1980s with the use of restriction fragment length polymorphisms (RFLP) to discover genes responsible for single gene Mendelian diseases such as Cystic fibrosis (12). In the 1990s, linkage analysis (using markers within families to map the location of genes associated with diseases) with microsatellites became the preferred approach, with genes identified for additional monogenetic diseases such as Huntington's disease (13). However, the lack of significant findings for common diseases using linkage analysis soon led to the hope that association studies might lead the way towards uncovering the genes involved with more complex diseases (14). Common diseases are an important research question for researchers in part because 9 out of 10 of the leading causes of death in the US are common diseases with some genetic component: heart disease, cancer, stroke, chronic lower respiratory diseases, unintentional injuries, diabetes, Alzheimer's disease, influenza and pneumonia, kidney disease, and septicemia (15). At that time, though, there was no map of markers that could be used in a genome wide association study. The Human Genome Project, which was completed in the early 2000s, provided the basis to create such a map (16). The International HapMap

project expanded upon the findings of the genome project and mapped out the patterns of linkage disequilibrium and haplotypes across the genome (17). These two breakthrough projects have provided researchers with a dense map of genome variations that, with advances in genotyping technologies, have made high throughput, high density genotyping possible. (18)

Single nucleotide polymorphisms (SNPs) are currently the preferred markers used for high density genotyping, although new probes like CNVs are also available. SNPs are a single base pair change in the genome. They are generally defined as a polymorphism with a greater than one percent population frequency of the rare allele and are the most common form of genetic variation in the human genome (19). Because SNPs are so common, and relatively easy and less costly to genotype than microsatellites, they have become the predominant type of polymorphism used in genome wide association studies.

Studies published on the genetic components of common and chronic diseases such as diabetes, heart disease, and stroke using SNPs are now underway with genome wide association studies (GWAS) being one of the primary method of investigation (see 2.2.2.). Densely spaced tagSNPs (SNPs which are highly correlated with a large number of other SNPs and thus, act as a proxy in genome wide association studies) are genotyped along the genome, and then a case-control or population cohort epidemiologic study design is used to measure the correlation between the alleles and the observed phenotype.

2.2 Genome Wide Association Studies and SNPs

Genome wide association studies (GWAS, see 2.2.2.) provide advantages over other statistical genetics approaches because they allow the researcher to measure genetic variation in unrelated individuals which allows for easier study recruitment, don't require a pre-existing biological hypothesis beforehand, use polymorphisms that span the entire human genome, and can detect smaller effects more readily than linkage studies (20, 21). Successful genome wide association studies need to be designed well and analyzed correctly, but even then the results are often modest and the functional significance of the results often remain in question (22). Valid studies need to ensure that cases and controls are selected accurately, that confounding issues such as ethnicity are addressed, that the genotype proportions are at Hardy Weinberg equilibrium, and that the results are consistent with similar studies (7, 23).

The number of genome wide association studies have been increasing at a steady rate (24), demonstrating the need for informatics solutions to both organize and make sense out of the results of these studies. As discussed in the introduction, a key step is moving from statistical associations to the beginnings of causal associations by doing functional annotation of SNPs. A selected list of genome wide association studies are presented in Table 2.1 as a demonstration of the range of diseases that have used a genome wide association approach to their study.

Estimating the number of genome wide association studies that have been completed is difficult – a search in PubMed for GWAS produces over 2,000 entries up to December 2008, but most of them are not true GWAS based on the previously described measures of valid studies (25).

Disease	Chromosomal region or Gene implicated
Age related macular degeneration (26)	1q32 (CFH) and 10q26 (LOC387715/ARMS2)
Atrial fibrillation (27)	4q25 (close to PITX2)
Asthma (28)	ORMDL3
Atopic eczema and asthma (29, 30)	Filaggrin
Breast cancer (31)	FGFR2, TNRC9, MAP3K1, LSP1, H19
Colorectal cancer (32, 33)	8q24.21
Coronary heart disease/ myocardial infarction (34, 35)	9p21 (close to p15/p16 CDKN2A/CDKN2B)
Crohn's disease (36)	ATG16L1, PHOX2B, NCF4, and a predicted gene on 16q24.1 (FAM92B), CARD15, IL23R
Type 1 diabetes (37)	12q24, 12q13, 16q13, and 18q11, CTLA4, major histocompatibility complex locus
Type 2 diabetes (38)	TCF7L2, CDAL1, CDKN2B, IGF2BP2, HHEX/IDE, SLC30A8, FTO, etc

Gall stone disease (27)	ABCG8
Multiple sclerosis (39)	IL2r, IL7-alpha, and ILA-DRA
Prostate cancer (40)	8q24 and 17q
Restless leg syndrome (41, 42)	MEIS1, BTBD9, locus between MAP2K5 and LBXCOR1
Rheumatoid arthritis (43)	6q23, major histocompatibility complex locus
Systemic lupus erythematosus (44)	IRF5

Table 2.1. Selected list of recent genome wide association studies (45).

2.2.1 Single Nucleotide Polymorphisms as Markers

As noted earlier, single nucleotide polymorphisms (SNPs) are the most common type of genetic variation in the human genome and are an important marker used in genome wide association studies (GWAS). Single nucleotide polymorphisms occur when there is a change in one base pair along the nucleotide sequence (Figure 2.1).

Single nucleotide polymorphism

ATTTC~~C~~CGGGAAGGAATATCCG
ATTTC~~C~~CGG~~G~~ATGGAATATCCG

Figure 2.1. Diagram of single nucleotide polymorphism, demonstrating a change in one base pair in a sequence of DNA.

There are approximately 11 million SNPs in the human genome, occurring about 1 in every 300 nucleotide bases (46). The largest repository for SNPs is dbSNP (47), which as of build 128, the newest version of dbSNP, has approximately 12 million uniquely mapped SNPs stored for humans, demonstrating that the database is approaching comprehensive coverage of SNPs (48).

Recent studies in GWAS have shown that SNPs are correlated to the phenotype variation for a number of common diseases (49). Studies have reported statistically significant results for a range of diseases ranging from breast cancer, heart disease, and diabetes, to prostate cancer as discussed in Table 2.1 (50). A recent catalog available of different genome wide association studies are available from the National Cancer Institute (NCI)-National Human Genome Research Institute (NHGRI)'s website (51) which as of December 2008 published 237 associations which assay at least 100,000 SNPs and report a p-value $< 1.0 \times 10^{-5}$ (25). Most SNPs don't contribute to phenotype outcome directly. Those SNPs that do contribute to phenotype outcome are generally located in the coding regions or regulatory regions of DNA (52). As described in the introduction, the process of predicting the biological outcome of a SNP is what we term "SNP annotation".

2.2.2 Definition of Genome Wide Association Studies

In a genome wide association study, researchers are able to extract millions of SNPs that are common in a population down to those SNPs that are linked to the phenotype or trait being studied in the GWA study (Figure 2.2). Markers in the form of tag SNPs are used as proxies in genome wide association studies. Tag SNPs are selected based on the concept of linkage disequilibrium, as measured by the correlation coefficient r^2 (53). The top section of Figure 2.1 (Figure 2.2, 1) demonstrates a group of individuals with markers that are common between them. A genome wide association study is conducted on a sample size large enough to detect any effects from the markers that are shared in the general population. The genome wide association study identifies those SNPs that are statistically associated to the phenotype, N stands for numbers of SNPs for (Figure 2.2, 2). However, although genotyping costs are decreasing and higher throughput platforms are always being developed (54), limitations in time and finances argue for the need to further limit the number of potentially functional SNPs that the researcher has to investigate and avoid any false positive results. This is what our system, SNPit, strives to provide.

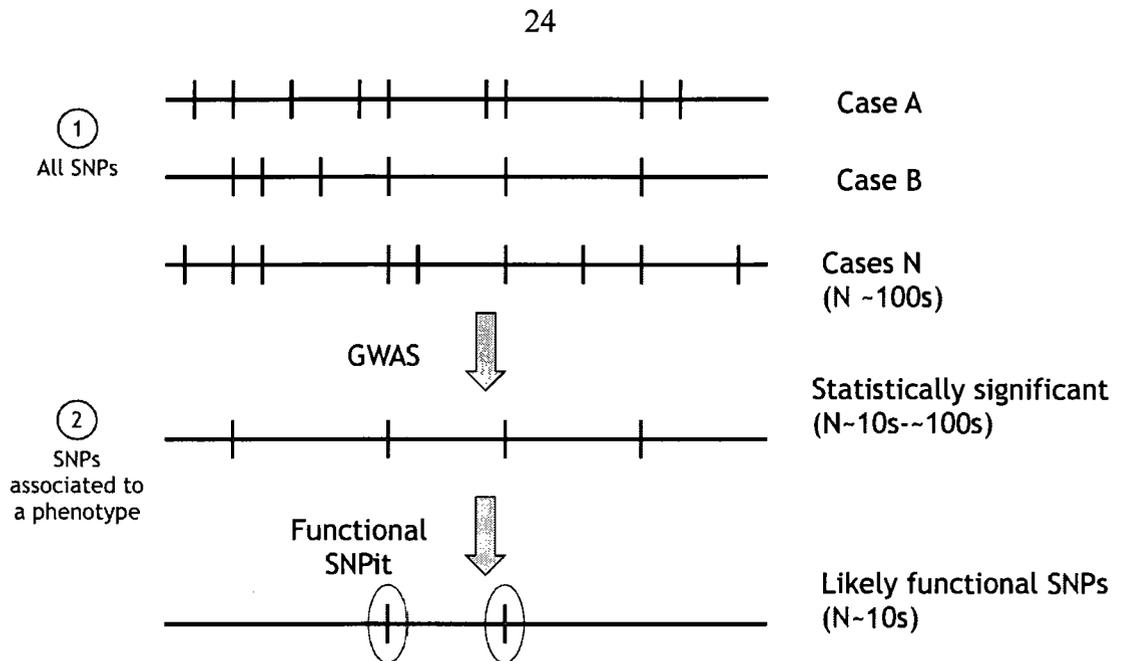


Figure 2.2. Diagram of common SNPs in a population of individuals and SNPs that are associated to the phenotype after a genome wide association study has been conducted.

One of the questions related to epidemiology is whether in a population, an association between a disease and an exposure can be demonstrated. Association studies ask the same question, only instead of an environmental exposure, it looks at a genetic exposure. The statistical approaches used in association studies are the same as that in traditional epidemiology, except applied to genotypes or alleles (6, 49). The analysis of the data in an association study depends on the study design, with follow-up studies using risk ratios and case-control type studies using odds ratios, both ratios measuring the likelihood of one group developing a disease over the other.

Association studies' calculations can be constructed on either an allelic or genotypic level (Figure 2.3). Odds ratios can be calculated using the traditional formula of cells ad/bc . Hypothesis testing with the null hypothesis that there is no frequency difference in the genetic exposure among the cases compared to the controls can be calculated using chi-square analysis (55). The p-value is defined as the probability of observing a value as or more extreme than what was observed by chance alone (6, 14), with the recommended threshold as being 5×10^{-8} to determine statistical significance (14).

	Cases	Controls
2/2	a	b
1/2	c	d
1/1	e	f
	Chromosomes	
Allele	Cases	Controls
2	$2a+c$	$2b+d$
1	$2e+c$	$2f+d$

Figure 2.3. Epidemiologic 2x2 or 2x3 table for case control studies, can be done either by genotypes or alleles at a single marker (20).

With the advances in high throughput genotyping techniques, it is possible to genotype hundreds of thousands of markers routinely, though high throughput genotyping is still costly (56). Many recent genome wide association studies have shown the potential for identifying disease causing variants. For example, the

Wellcome Trust Case Control Consortium published a study using the Affymetrix 500K GeneChip to study seven different diseases in 14,000 individuals (36).

Over the past couple of years, genetic epidemiology studies have shifted from Mendelian diseases to polygenic, common diseases such as Alzheimer disease (OR = 4.01) (57), type 2 diabetes (OR = 1.10 and 1.25) (38), Crohn's disease (OR = between 1.14 and 1.63) (36), bipolar disease disease OR = (between 0.84 and 1.84 p) (36), stroke (moderate-high with odds ratios ranging from 0.40 to 0.54 and 1.9 to 8)(58), and coronary heart disease (OR = between 1.20 and 1.33) (35) . Results have usually been shown to be modest, with odds ratios above 1.5 being uncommon.

2.2.3 Genotyping and statistical analysis tools

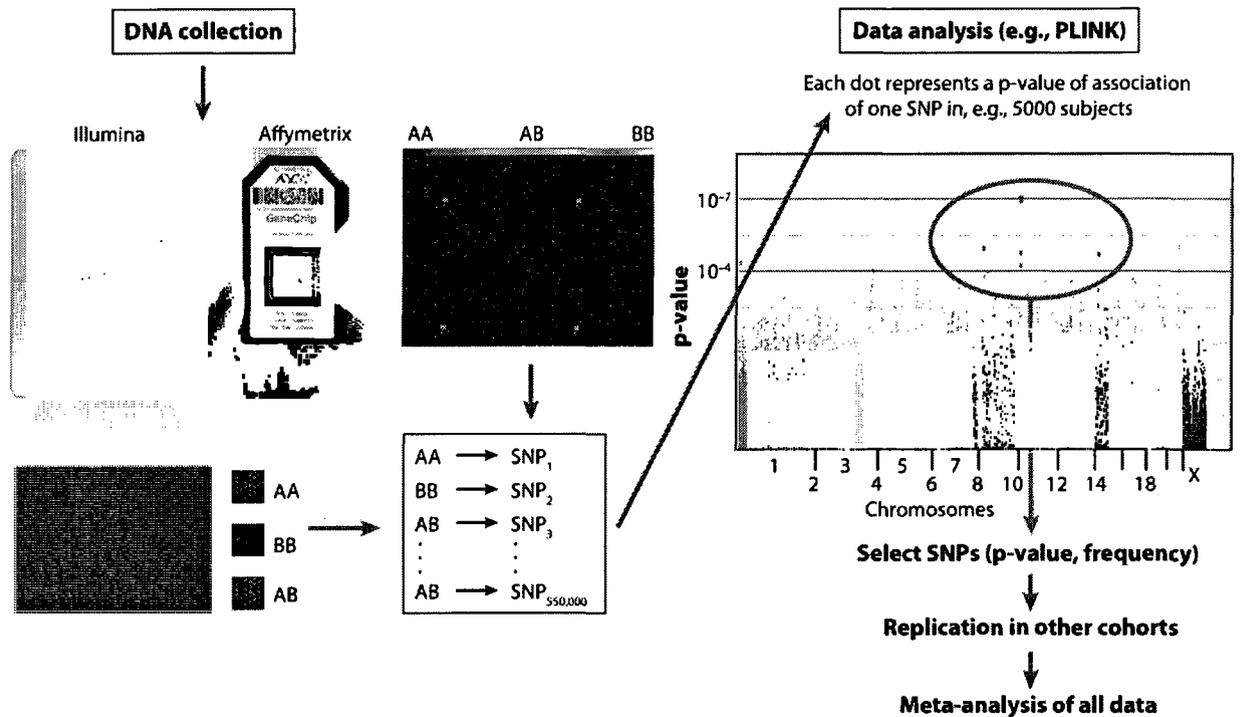
SNP microarray genotyping is the most common type used in genome wide association studies, though there are many different techniques that researchers can choose from. The main producers of SNP chips are from Illumina and Affymetrix (56). Affymetrix's Genome-Wide Human SNP Array 6.0 includes probes for 906,600 SNPs and Illumina's High Density Human 1M-Duo chip probes more than 1 million SNPs (59). The main difference between these two chips is in how they select their SNP probes. Illumina uses haplotype tagSNPs based on findings from the International HapMap project, whereas Affymetrix only uses tagSNPs on half of

their platform while using random SNPs that cover the genome uniformly on the other half (59).

An illustrative hypothetical case control study would be comparing the genotype of patients with Type I diabetes to health controls. The investigator would identify a sample of patients with Type I diabetes by first identifying a standardized definition of disease and recruiting participants for the study. The investigator would identify a matched cohort of healthy controls by again recruiting for participants. DNA samples would be obtained from the participants. They would obtain genotype data using a given SNP platform (e.g. Illumina vendor High Density Human 1M-Duo chip probe chip). In consultation with biostatisticians the investigator would choose among different types of statistical programs, such as PLINK, to calculate the statistics of their sample. The X^2 value is then converted to a p-value in PLINK and presented in a dot plot (Figure 2.4).

SNP microarrays use the same principles of probes and hybridization as DNA microarrays. Each SNP chip typically corresponds to one sample, with a dye color assigned to a particular allele. Depending on which platform is used, the samples are hybridized to the SNP chip and the ratio of fluorescence is measured for the genotype (AA/BB/AB). PLINK is a tool that can help researchers to analyze the data and highlights which SNPs have the lowest p-value in the entire subject population. Once a statistical hit is found by an investigator for a particular SNP, the hypothetical investigator would have an odds ratio and p-value for Type I diabetes

which when functionally annotated might prove to be a coding region non synonymous mutation of a gene involved with a potential biological pathway, or it might just be a tagSNP.



 Lamberts SWJ, Uitterlinden AG. 2009. Annu. Rev. Med. 60:431–42

Figure 2.4. Illumina and Affymetrix SNP microarray platforms (60).

2.2.4 Why are SNPs used in genome wide association studies

As previously stated, the genetic hypothesis behind SNP association is that cases will carry the risk genotype more frequently than controls, which would mean that there is a statistical association between the cases and controls (49). Genome wide association studies (GWAS) take advantage of this concept and compare tens of thousands of SNPs across the genome among unrelated individuals. GWAS then use epidemiologic statistics such as odds ratios and risk ratios to measure the difference in allele frequencies between the case and control groups, as discussed previously in section 2.2.2.

GWAS have the additional benefits of not requiring family data and detecting smaller effects in a study. For genome wide association studies, researchers can be even more all encompassing without a prior hypothesis or genetic location in mind. This is what differentiates it from a candidate gene approach where researchers look at specific genes based on previous results or prior biological information. Because it is more exploratory, genome wide association studies have been touted as being able to uncover the genetic mechanisms behind common diseases (50). However, the reproducibility of genome wide association studies have been low in the past (61), thus, most published GWA studies now require that replicated results be included in submissions (62).

The previous low reproducibility in genome wide association studies can be due to a variety of factors. The design of the study might be flawed, which would lead to study results that aren't valid. The control group needs to conform to Hardy-

Weinberg distribution, checking for HWE in the genotype frequencies help to ensure that there is no confounding in the sample (7). Multiple comparisons when conducting statistical analysis and population stratification during the selection of cases and controls help to avoid false positives. Even if the genome wide association study is well designed, the SNP results from GWA studies need to be replicated in independent populations and also followed up with biological explanatory information in order to support either a true direct or indirect association.

2.3 Challenges to applying results of GWA studies

The practical applications of genome wide association studies are still uncertain because there is often a lack of causal association and/or lack of preventive measures and/or lack of therapeutic measures. Furthermore, for a given phenotype often different studies show statistically significant association with little overlap of SNPs across studies (the problem of lack of reproducibility). For example, studies on Parkinson's disease have demonstrated difficulties in replicating results perhaps due to sample size (63).

Nevertheless, there are examples in which concrete applications can be considered. For example, a SNP along chromosome 9p21 has been found to increase the risk of myocardial infarction independent of traditional risk factors, this kind of additional information may be useful for risk prediction purposes (64) identifying a group for

whom traditional preventive measures for other risk factors may be especially important.

Once a genome wide association study has been completed and publicized, expectations are that they will have some kind of impact on public health and clinical practice. However, the exact form of that impact has yet to be determined (60, 65-67) due to some of the challenges alluded to earlier. There are many exciting possibilities when it comes to the application of genome wide association studies in areas such as genetic testing (45, 60, 65-68), pharmacogenetics (1, 2, 60, 69, 70), and public health surveillance (1, 60, 66, 71, 72), however, before the value of genome wide association studies can be actualized, the challenges to their use must be addressed.

2.3.1 Contrasting clinical and statistical significance

A key challenge to using information from GWAS studies is the difference between statistical and clinical significance. The summary results reported in a genome wide association study only provides information on statistical significance. The chi square summary statistic provides an observed value which is then compared to what would be expected if replications of the study was done with the assumption that the null hypothesis is true (that is that there is in fact no difference in genotype

between the case and the control groups). If the observed value is very unlikely to have occurred by chance alone, we deem the p-value to be statistically significant, usually at the widely accepted cutoff of 5×10^{-8} (accounting for the number of tests using the Bonferroni correction). However, because it takes a very large number of samples in order to attain this level of statistical power, genome wide association studies often take a multi-staged approach, with more liberal p-values allowed during the first stage of a genome wide association study, even though this can lead to more false positive results. In the second and third stages, the p-values would be adjusted to more stringent levels. (8). For example, multiple association tests were conducted along the genome for rheumatoid arthritis, the values were plotted as $-\log_{10}$ p-values; two different thresholds, $P < 10^{-4}$ and 5×10^{-8} are shown to demonstrate which SNPs attain statistical significance (Figure 2.5).

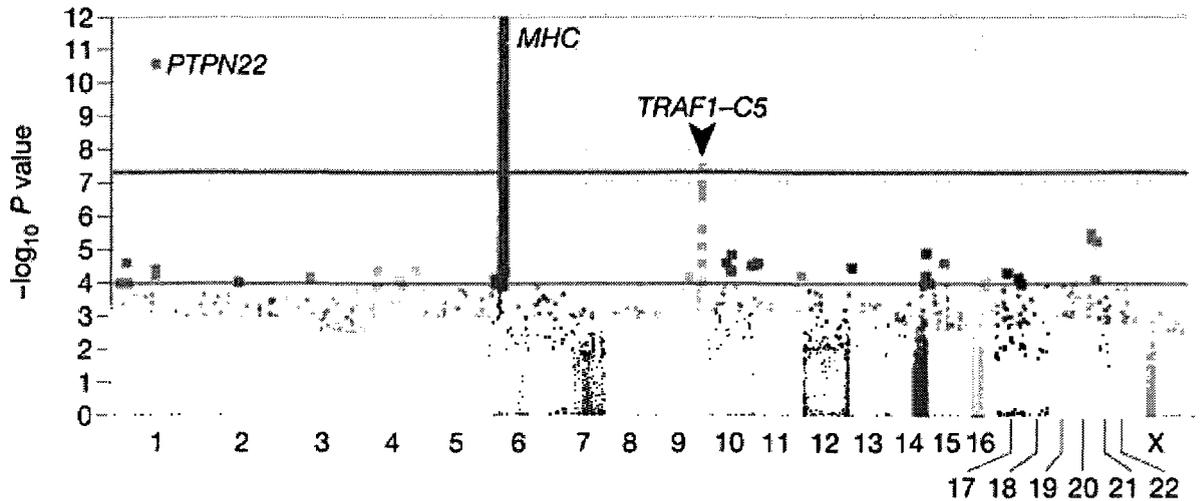


Figure 2.5. Genome wide association study results for rheumatoid arthritis (73).

Once a statistically significant measure is achieved, though, is that result of any clinical significance (also known as clinical utility). In other words is the measure of interest and relevance to the patient and their clinician? A very small difference (RR near 1) can be statistically very significant but if other risk factors are present that have a much higher RR, then that difference might not be clinically significant. In order to be considered clinically significant, a study needs to both achieve a statistically significant outcome as well as an outcome that has clinical importance (74). For example, a recent study on adult and childhood height found a statistically significant SNP in the HMGA2 gene with a p-value of $P = 4 \times 10^{-8}$, this variant was then replicated in additional studies as well (75). However, it was estimated that this particular variant only accounts for 0.3% of the population's variation in height, which might not be considered a clinically significant outcome.

Furthermore, studies on traits such as height might not be considered as clinically important as studies on human diseases associated with increased morbidity and mortality, particularly where the RR is relatively large. Thus, the HMGA2 gene is an example of a GWAS that is statistically significant, but almost certainly not clinically significant. Genetic diseases with high penetrance as well as treatment and screening benefits such as multiple endocrine neoplasia type 2, hereditary breast ovarian cancer syndrome and familial hypercholesterolemia are examples of conditions that have high clinical significance and utility (76). The important point is that statistical significance without clinical significance is of limited practical application. In addition, the risk of false positives means that statistically significant GWA studies need to be validated.

Related to the question of clinical significance, there is also the need to examine whether or not a GWA study is clinically valid and possesses clinical utility. Clinical validity refers to the issue of whether or not the marker being used in a genome wide association study correctly classifies the cases from the controls for the trait being studied (77). Clinical utility has a wide spectrum of definitions, though most commonly it refers to the costs and benefits of introducing a genetic test into practice, particularly its health outcomes (78). For rare phenotypes even low rates of misclassification may result in far more controls being incorrectly classified as having the trait than actual cases classified as having the trait. It has been recognized that the receiver operating characteristic (ROC) curve is the most suitable option for

evaluating the clinical validity of genetic tests (79). ROC curves plot the test's true positive rate against its true negative rate over a range of cutoff points score statistics. ROC curves is also one of the most popular ways to evaluate clinical diagnostic tests and is used in a wide number of medical disciplines (80). The area under the curve (AUC) is a summary index of a ROC curve, an example of this approach can be seen by plotting the predictability of three common variants for the risk of type 2 diabetes (Figure 2.6). A ROC curve with the same predictive ability as that which occurs by chance would be a straight diagonal line starting from the bottom left corner (with an AUC of 0.5). A test that performs perfect prediction would be visualized as a ROC curve that goes straight up the y-axis and then straight across the x-axis (with an AUC approaching 1.0). The AUCs in figure 2.6 would not be considered to be very good as compared to most non-genetic testing in widespread clinical use where the AUC is generally 0.95 or greater.

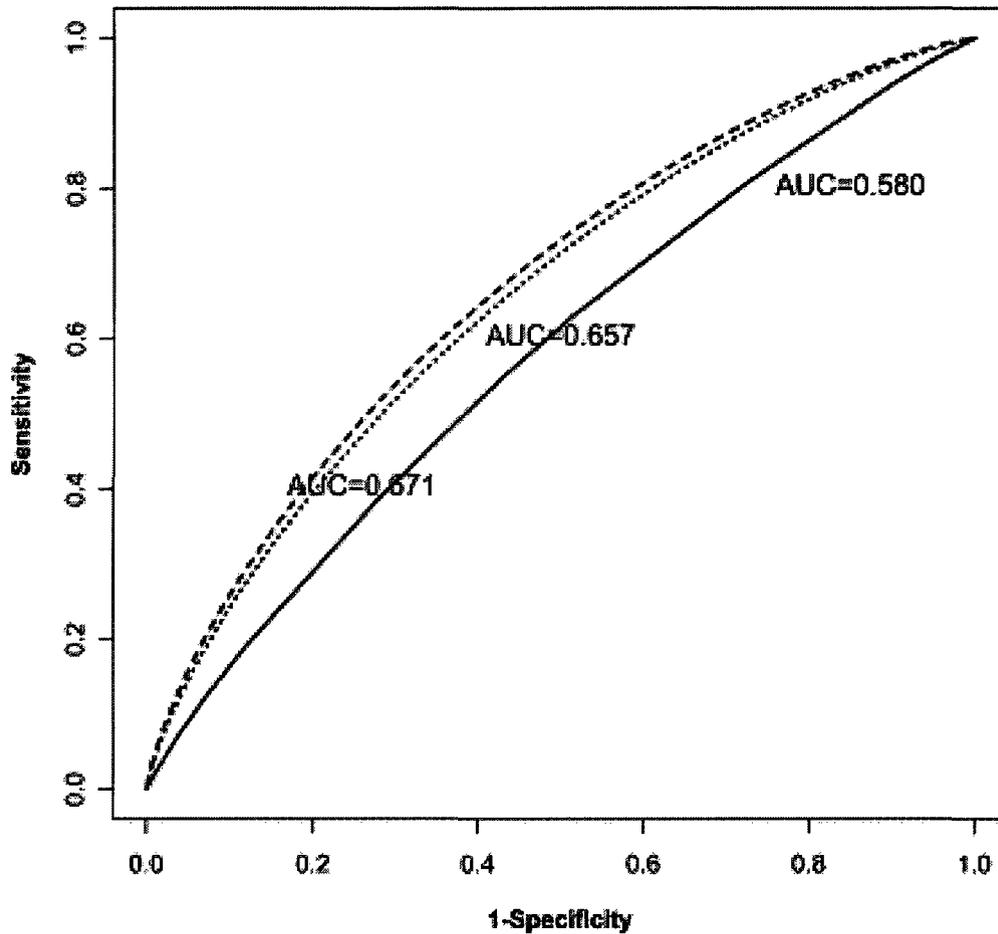


Figure 2.6. ROC curves for 3 Type 2 diabetes predictive tests (80).

Clinical utility, on the other hand, refers to what can actually be done if an individual is found to be at risk for the disease (60). For diagnosis and screening clinical utility depends on whether or not there are preventative measures that can be taken in both the general population as well as the genetically at risk cohorts (77). For example, Duchenne's muscular dystrophy provides a prediction rate of 100% in

genetic testing, and Huntington's disease testing in families provides a 95% prediction rate (60) demonstrate genetic tests currently being used that have high predictive value, but somewhat limited clinical utility since as yet there is no cure for these diseases. The clinical utility that results from these tests revolves around the ability of those tested to make life planning decisions and decisions around having children.

Clinical validity and clinical utility are inter-related, and it is not yet clear what kind of impact a genome wide association study will have in terms of these two measures. Even for monogenetic diseases, the biological path from SNP to protein is far from clear and differences in the clinical validity of classic monogenetic disease still occurs. For example, familial hemochromatosis is currently being considered for screening since the HFE gene and two associated SNPs have been found to be associated to the disease (60). Recent studies, though, have since recommended against screening for familial hemochromatosis because HFE penetrance has an uncertain development timeline even when no intervention is undertaken (81).

The clinical utility of genetic markers for risk of common diseases such as diabetes, cancer, and heart disease is currently even more limited because genetic factors for these diseases are complex. Additionally environmental and lifestyle factors may play an even larger role in common diseases than genetic factors; for example, even though BRCA1 and BRCA2 genetic testing for breast cancer is already available, the mutations only account for 3-10% of breast cancer cases (65).

Another example of this is coronary heart disease, although recent findings near chromosome 9p21 identified a variant that increases the risk of CAD separate from other risk factors (OR > 1.20) (64). However, traditional risk factors account for 80% of the population's risk to coronary heart disease (82) such as age, sex, body mass index, hypertension, and smoking (83). The 10q11.21 locus, another variant example, showed a significant effect in women (OR=1.29 [1.15 to 1.45], $P=1.86 \times 10^{-5}$) but not men (OR=1.03 [0.96 to 1.11], $P=0.387$) (83).

Even for a genetic test with 100% clinical validity, the issue of clinical utility needs to be addressed. Translating the results from genome wide association studies into clinical practice will depend on whether or not any medical prevention or intervention steps can be taken, as well as the public's perceptions concerning genetic testing (84). For example, there is currently evidence that a mutation in the APOE4 gene leads to an increased risk in Alzheimer's disease (85), however, there is no current treatment for the disease. Furthermore, the predictive value or clinical validity of the APOE4 gene is questionable since those without the allele mutation might still develop the disease (65).

For a genetic test to be clinically useful, it needs to demonstrate predictive value as well as some kind of medical and/or public health benefit. There will also need to be public education and a modification in policy when it comes to public perceptions. These kinds of public education and policy modifications will require insights on the legal and ethical consequences of genome wide association studies.

2.3.2 Law and Ethics

The social, legal, and ethical implications of genomics research are enormous and consequential (86). The results from genetic testing and screening bring about numerous ethical questions. A full discussion of these issues is outside the scope of this dissertation. Three key issues are highlighted very briefly here.

- Is the data related to genome wide association studies databases properly de-identified? Can genomic data truly be completely de-identified? Have all possible measures in terms of protecting the individual been taken? (87)
- The traditional forms for informed consent might not work for studies related to genome wide associations. Do the study participants really understand where their information will be stored and perhaps distributed in the long run? What should be done if they choose to withdraw from the study in the future, particularly if the data resides in public databases? What if the participants need to be contacted concerning the study? Do the participants understand all the risks and benefits when signing a consent form? (87-89)
- The identification and formation of an independent body that can govern and monitor the study and study results that pertain to genome wide association studies needs to be created. This governing body would depend on the exact

study being conducted and would need to include ethics experts to deal with the long range ethical issues that could arise. (87, 88)

Additional legal and ethical issues range from employment and insurance genetic discrimination, health disparities, and legal policies are all issues that need to be examined and kept in mind when conducting any kind of genomic research (86, 88, 90). ELSI issues related to the work of this dissertation concerns false positive results that might lead researchers to focus on the wrong region.

The main ethical and legal issues identified above as related to data sharing do not yet have definitive answers because these are issues that are still in the beginning stages of being identified, discussed, and debated. However, the main take home lesson as related to this dissertation is that these issues as well as those related to clinical utility and validity as discussed in section 2.3.1 need continued examination if the vision of predictive, preventive, and personalized medicine is ever to become a reality.

2.3.3 Applications of GWA studies

In this next section, we will examine how results from genome wide association studies are already being applied in actual clinical and public health settings such as through pharmacogenetics (69), genetic testing (60), and genetic screening (86). Related to genetic testing we will address direct to consumer (DTC)

genetic testing since it can already be purchased over the internet, and through a medical doctor, and both population and prenatal screening is already occurring (90, 91). The National Office of Public Health Genomics at the Centers for Disease Control and Prevention has established the Evaluation of Genomic Applications in Practice and Prevention (EGAPP) Initiative to establish guidelines for evaluating genetic tests based on analytic validity, clinical validity, and clinical utility (92). These and additional applications of GWA studies are examined in this section. As briefly discussed in section 2.3.1.3 the complete ethical and legal implications of this testing is outside of the scope of the introduction to this dissertation.

2.3.3.1 Pharmacogenetics

The application of the results from genome wide association studies to better understand the efficacy of drugs is the aim of pharmacogenetics (also called pharmacogenomics) (69). Numerous genetic variations have been found to result in adverse drug reactions (ADRs) (69). These variations can reside along any genes that code for proteins related to drug metabolizers, drug transporters and receptors, or any other molecules that play a part in the pathways related to drug efficacy (93). Genome wide association studies can potentially help with a subset of ADRs that are due to therapeutic drugs and responses they incur (93). For example, variations in both the CYP2C9 drug metabolic enzyme and part of the vitamin K pathway have

been found to effect the dosage requirements of the anticoagulant warfarin (60, 70, 94). This difference in dosage requirement due to genetic variants was substantiated enough for the Food and Drug Administration to change the drug label of warfarin (95). Numerous genome wide association studies are currently being conducted to uncover the genetic contributions to variations in drug responses (69), demonstrating the potential for using genome wide association study results to better understand drug efficacy.

Table 2.2 lists some of the recent discoveries demonstrating genetic variations and their effects on adverse drug reactions, taken from the 'Table of Valid Genomic Markers in the Context of Approved Drug Labels' on the US Food and Drug Administration website (95). It is worth noting that even for warfarin, at the time the FDA made its recommendation, there was no algorithm or accepted testing approach to tailor treatment. Since then an algorithm factoring in genotype for warfarin dosing has been published (96) based on retrospective data and a prospective study of this equation is currently underway.

Gene	Drug	References
(1) Tests recommended by Food and Drug Administration		
CYP2C9	Warfarin	Anderson et al. (97)
VKORC1	Warfarin	D'Andrea et al. (98)
TPMT	Azathioprine	Snow and Gibson (99)
UGT1A1	Irinotecan	Innocenti et al. (100)
HLA-B*1502	Carbamazepine	Chung et al. (101)

HLA-B*5701	Abacavir	Mallal et al. (102)
------------	----------	---------------------

Table 2.2. List of genes that influence adverse drug effects (93).

2.3.3.2 Direct to Consumer Genetic Tests

Direct to consumer genetic tests are already available for a wide range of common diseases from diabetes to Alzheimer's disease (Table 2.3) (45). However, the clinical validity and clinical utility of these tests have yet to be proven and there is a serious lack of regulation in this industry (67). Patients often do not understand what the results of genetic epidemiology studies really mean. For example, current genetic tests for breast cancer describe an increased risk of 1.6-fold for individuals with two copies of risk allele, but the absolute risk would only increase from 3% to 4% (103).

As mentioned previously in section 2.3.1, the statistical effects of alleles reported in genome wide association studies are currently very modest. Federal regulation currently does not extend to genetic tests done in laboratories (104). Table 2.3 lists out the companies that offer tests directly to the consumer for a wide range of diseases ranging from diabetes to heart disease (Table 2.3).

As more and more genome wide DTC testing is becoming available (such as that offered by 23andMe (105), deCODE (106), see Table 2.3) the scale of the problem is increasing. Because of all the previously discussed limitations of genome wide association studies (section 2.3.1), the results from direct-to-consumer genetic tests are far from clear. The lack of regulations of these genetic tests are a serious concern, so much so that the states of New York and California have decided to enact their own regulations dealing with the need for laboratories that offer these tests to have a license and requiring that a potential customer have a physician's order to purchase one of these products (104). The argument that these consumer tests are purely for health information purposes and perhaps genealogical information is also being debated. The benefits and harms of direct-to-consumer tests is something that points to the need for regulation and oversight, the EGAPP guidelines as mentioned in section 2.3.3 offers recommendations for evaluation of these genetic tests.

Company (base country)	Claimed prediction	Market
23andme (USA)	Whole genome scan: common diseases	USA
deCODE (Iceland)	Diabetes, atrial fibrillation, myocardial infarction	USA
deCODE (Iceland)	Whole genome scan: common diseases (deCODEme)	Global

GenoSense (Austria)	A range of common diseases	Europe, Canada, USA
Genetic Health (UK)	Offers GenoSense's tests	UK
Geneticom (Netherlands)	Range of common diseases	Europe
Genovations (USA)	Cardiovascular disease, osteoporosis	USA
Graceful Earth (USA)	Alzheimer's disease	USA
Interleukin (USA)	Heart disease	USA
Mygenome.com (USA)	Cardiovascular disease, osteoporosis, Alzheimer's	USA
Myriad (USA)	BRCA (breast cancer); Melaris (skin cancer)	USA, Europe
Navigenics (USA)	A range of common diseases	USA

Table 2.3. List of different genetic tests currently available to the public (45).

2.3.3.3 Public Health Surveillance

Genetic screening and newborn screening are both forms of public health surveillance. If we take the general definition of public health surveillance as the collection and use of health related information (107), then genetic screening and

prenatal screening can be seen as forms of surveillance. Genetic screening, defined as the search for asymptomatic diseases in individuals, has been increasing in numbers for the past 50 years (86), with the general view that genetic screening can be used for prevention purposes (71). Similarly, newborn screening, has been increasing in numbers of diseases, with screening now being done for more than 50 diseases (91). Thus, genome wide association studies contribute to public health surveillance.

2.3.3.4 Elucidation of pathogenic pathways

One of the main challenges of genome wide association studies involves understanding the biological impact of a polymorphism found to be statistically significant might be. In addition to providing biological background information and addressing the functional annotation of the genetic variants found to be significant in genome wide association studies, research into functional and regulatory networks is necessary to understand the biological mechanisms behind the markers in GWA studies (1). For example, genetic risk factors to type 2 diabetes involve genes related to pathways involved with the creation of B-cells in the pancreas and also in pathways related to how glucose is metabolized (108, 109). Single nucleotide polymorphisms act as markers in the genome wide association studies that help to identify the variants in these pathways, elucidation of pathogenesis requires

additional laboratory research and research into the biological mechanisms involved with the SNPs found to be statistically significant in a GWAS.

2.3.4 Stakeholders in understanding role of genetic variation

Stakeholders are individuals, groups, or organizations that have some kind of interest in the conduct and application of genome wide association studies.

Researchers that conduct studies on genome association studies carry out the GWA study and publish the results. At the clinical level the results of genome wide association studies are important to physicians and consumers if they are translated into clinical testing for diagnosis, prevention, treatment or prognosis of disease (66). Direct to consumer tests are already available for everything from breast cancer, Alzheimer's disease, heart disease, and genome wide scans (45).

Public health departments where public health officials use the results of genome wide association studies play a role in GWA studies, for example, recent proposals have been to include breast cancer screening in public health programs (110). Private and public hospitals where doctors assess and relay information to patients on genome wide association studies also have a stake in GWA studies; for example, growing knowledge on the genetics of type 2 diabetes as well as more awareness of the benefits of understanding the molecular mechanisms of diabetes

have led clinicians to become more aware of the potential to use molecular genetics to improve patient care (68).

Finally, at the national level, research at different institutions such as the National Institute of Health, the Centers for Disease Control and Prevention, and the department of Veteran's Affairs are also stakeholders in GWAS results might influence the decisions made by funding agencies in terms of prioritization in research funding. Recent figures have demonstrated that genomic research funding has increased steadily throughout the years in many countries around the world, with the US leading in genomic research spending from 2003 to 2006 (111).

The range of stakeholders who have some interest in the results of genome wide association studies make the application of these results to be that much more significant.

2.4 Discussion

In this chapter, we presented an introduction to genome wide association studies and SNPs, to set the stage for the need for functional annotation. We then examined the difference between statistical and clinical significance. The applications of genome wide association studies are still in their beginning stages of exploration and the broader clinical application of these genetic discoveries are currently unclear though there are beginning to be applications in the form of pharmacogenetics, genetic screening, and genetic testing. In addition, even though

genotyping technologies have been advancing, researchers need to prioritize their time and resources, as well provide biological explanatory support to their GWAS results. This need requires more exploration into the biological mechanisms that drive SNP functionality, and this exploration and the challenges to functional annotation are discussed in Chapter 3.

In the next chapter, we will be discussing the potential biological mechanisms impacted by SNPs. Genetic variations can lead to a change in the final outcome through a variety of methods. These methods comprise mainly from determining the location on which the SNP resides. Most SNPs (95%) fall into non-coding regions along the genome and are thought to be less likely to affect protein function (112), those in areas that are close to or within genes are thought to be more likely to affect protein function. SNPs can alter any of the steps from transcription, when the DNA is turned into RNA, through translation, when the RNA is turned into protein, and even affect post-transcription.

Nonsynonymous SNPs are thought to be the most damaging type of SNP variation. They can either be a missense mutation, where the SNP causes the amino acid to change to a different codon, or a nonsense mutation, where the change in SNP causes a premature stop codon. While nonsynonymous SNPs affects the amino acid sequence, synonymous SNPs and intronic SNPs can impact gene expression by disrupting the regulatory or splicing region of a sequence (113).

Chapter 3: Current Approaches and Challenges to Functional Annotation of SNPs

3.1 Introduction

As previously discussed in section 2.3.1, a SNP that is statistically associated with a phenotype may or may not be of clinical significance or utility. Functional annotation of a collection of SNPs associated with a phenotype can provide insight into potential biological mechanisms of causation particularly if it appears that the SNP is a mutation that might have functional impact on the regulation of protein transcription or on the amino acid sequence of the protein itself. The location of a given SNP can impact gene regulation, transcription or translation and in turn one can infer likely functional impact (21, 93, 114, 115). Broadly, SNP annotation is done either manually or using automated/semi-automated systems. Limitations to current approaches led to the overarching question of this dissertation which is to determine the feasibility and value of federated data integration with combinations of logical and probabilistic inference for SNP annotation. Even once a SNP is fully functionally annotated the issues of clinical significance and utility discussed in Chapter 2 remain.

3.2 Types of Functional SNPs

Most SNPs (95%) fall into non-coding regions along the genome and are thought to be therefore less likely to affect protein function (112). Those in areas that are close to or within genes are thought to be more likely to affect protein function or expression (116). SNPs closer to genes or within genes can alter any of the steps from regulation of rate of transcriptions, to transcription, when the DNA is turned into RNA, through translation, when the RNA is turned into protein, and even affect post-transcription.

Nonsynonymous SNPs are thought to be the most damaging type of SNP variation. They can either be a missense mutation, where the SNP causes the triplet codon for an amino acid to change to a different codon, or a nonsense mutation, where the change in SNP causes a premature stop codon, occurring only at a frequency 1.49×10^{-4} in dbSNP. While nonsynonymous SNPs affects the amino acid sequence, intronic SNPs and regulatory region mutations can impact gene expression by disrupting the regulatory or splicing region of a gene (113).

3.2.1 SNPs Involved with Transcription

SNPs can impact the transcription process of copying a sequence of DNA into the appropriate mRNA, at multiple stages (Figure 3.1). During the pre-transcriptional processes, promoters that reside a couple of base pairs upstream from

the site of transcription help to initiate the process by directing RNA polymerase to bind typically through a TATA box sequence. Thus if a SNP resides in the promoter region of the DNA, that SNP might impact whether or not transcription is initiated or the efficiency of transcription. Similarly, there are enhancers that are around the transcription area, some even in intronic or intergenic regions, that can increase (or decrease) the level of gene expression that is transcribed with concomitant impacts of protein levels. Transcription initiation uses RNA polymerase as an enzyme, whether or not the RNA polymerase binds to the promoter depends on proteins called transcription factors, the DNA site on which the transcription factor binds to is called the transcription factor binding site. Thus if the SNP resides on a transcription factor binding site, that can impact the affinity of the transcription factors bind onto the DNA sequence, which would then affect the RNA polymerase binding to the promoter region again resulting in misregulation of protein expression. (116)

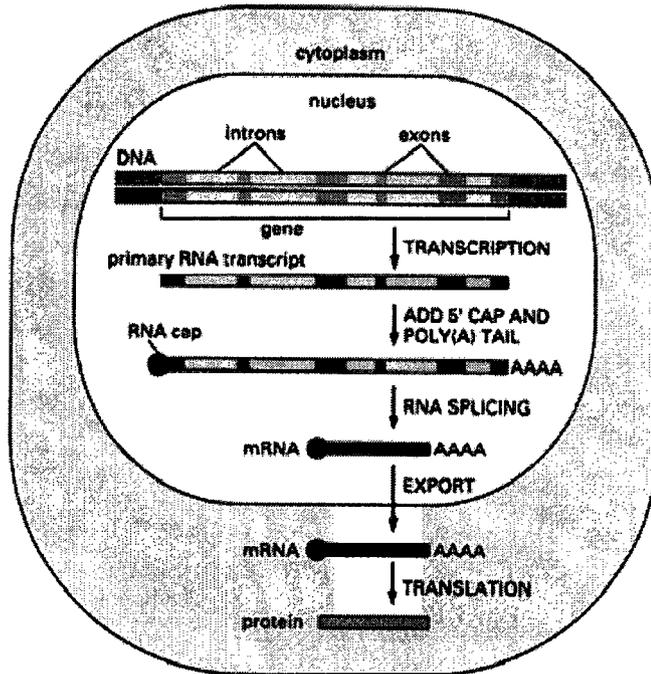


Figure 3.1. General diagram of the process of transcription and translation (117).

Once the primary transcript RNA is made, it needs to be turned into mature mRNA, this is where post-transcriptional modifications take place. During post-transcriptional modification, the introns are spliced out and the exons are connected to form the final mRNA. If the SNP resides around the splice site area, it can cause the final mRNA to change, resulting in alternative splicing with concomitant impacts on protein sequence. SNPs that reside around the UTR areas of the mRNA can also impact gene expression, because there are various types of regulatory sequences that can be found in both the 5' and 3' UTR regions.

3.2.2 SNPs Involved with Translation

Once the mature messenger RNA is made, translation takes place in the cytoplasm, where ribosomes and transfer RNA helps to transform the mRNA into a chain of amino acids. SNPs that might impact the post-translational formation of the protein include those that demolish protein function, such as nonsense SNPs, which changes the amino acid to a premature stop codon. SNPs that cause a frameshift in translation could also demolish protein function. Similarly, nonsynonymous SNPs in general can alter the protein function in more subtle ways. Also as noted earlier SNPs around splice site could result in alternate splicing resulting in different protein structure.

3.2.3 SNPs Involved with Gene Regulation

Gene regulation can occur during any of the steps necessary to turn DNA into protein. One of the more common forms of gene silencing is DNA methylation, when there is a chemical modification to the DNA (118). Regulation at the transcriptional level can also occur through alternative splicing. Regulation at the translational level can occur through enzymes that either add to or take away from the protein. SNPs related to gene regulation are probably the least understood out of the three categories.

3.2.4 Challenge to capturing information on alternative splicing

It is important to understand alternative splicing though, because exonic silencer enhancers and silencers can reside along both the intronic and exonic regions of a sequence. These enhancers and silencers of splice sites can both regulate and potentially demolish the final protein product (Figure 3.2).

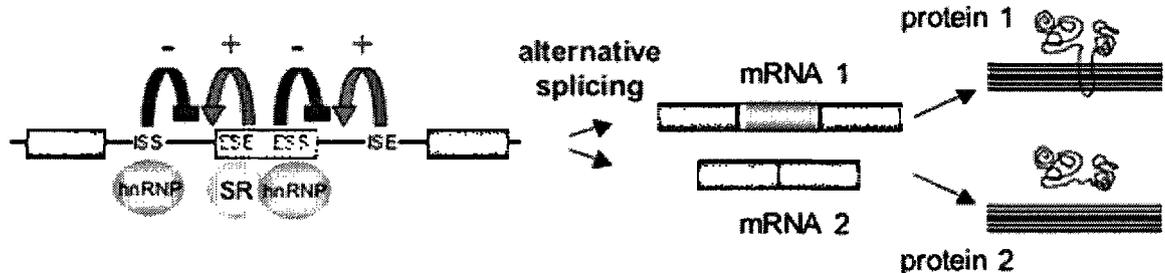


Figure 3.2. The mechanics of alternative splicing, silencers and enhancers in both the introns and exons influence the different ways that the exons can be spliced together. (119)

There are very few systems available that attempt to predict the exonic splicing enhancing and silencing effects on a SNP. ESEFinder is one of the few that looks at using sequence motifs to try and predict possible exonic splicing enhancer sites (120). However, even the ESEFinder system states that the high scores returned by their system doesn't necessary mean that the input is within an enhancer region. Furthermore, the ESEFinder system currently only provides results for four serine/arginine-rich proteins and their results don't compare across different proteins.

3.3 Manual Approaches to SNP Annotation

As introduced in Chapter 2, researchers want to annotate the results of genome wide association results because it helps to uncover the pathogenesis of the SNP as well as any additional genes or pathways that might be associated with the SNP. The majority of functional annotation on SNPs done nowadays is through manual or largely manual methods. Manual approaches to SNP annotation require first a knowledge of the kinds of databases that are available that provide information on the possible biological mechanisms of the SNP as well as understanding of how the various annotations that are retrieved one at a time might work together. The process of manually accessing each one of the relevant data sources, then manually downloading and storing the information, then manually analyzing the information is both time consuming and laborious. This manual approach, thus, does not scale up when researchers are dealing with large collections of SNPs. The larger the collection of SNPs that needs annotation, the more complex the search strategy becomes, and thus, the more potential for missing relevant results. In theory, the same process automated through a computer program or via semi-automated approaches saves both time and resources. Automated SNP annotation also eliminates human error that might occur when annotating the SNP manually due to failure to fully and systematically evaluate the entire search space.

3.4 Automated and Semi-Automated Approaches to SNP Annotation

The first attempts to use computational analysis to understand genomic variation can be traced back to around 10 years ago (121). However, SNP systems that integrate biological information on SNPs, classify variation by biological importance, and try to predict the functional annotation of SNPs have only occurred in the past few years.

Previous automated approaches to SNP annotation have been to use a data warehouse approach and to focus in on certain categories of SNPs, such as nonsynonymous SNPs. For this review of previous systems, the literature review is limited to annotation systems that include data integration of some kind and are published in the literature. Systems that use machine learning techniques to classify SNP effects were not included in this literature review. We choose not to examine machine learning techniques because we felt that the category of machine learning for classification purposes was not adequate for the purposes of functional prediction as there is not an adequate set of functionally validated data sets to train the machine with. Systems that are no longer working or not available for use were also not included in this literature review.

3.4.1 Review of current approaches to SNP annotation

Various previous systems and their strengths and limitations are displayed in Table 3.1. The majority of previous systems looked at annotating primarily nonsynonymous SNPs, such as LS-SNP, PolyDoms, and SNPs3D. Some of the systems looked at a wide range of possible predictors for SNP functionality, such as FastSNP and F-SNP which looked at a variety of transcription and translation mechanisms and its effects on SNPs.

System Name	Focus	Limitations	Data Integration Type	Evaluation Type
FastSNP (122)	Changing amino acids, transcription factor, splicing	No mediated scheme, use of minor allele frequency as validation	Uses web wrappers	Case study, looking at allele frequencies
F-SNP (123)	splicing, transcription, translation and post-translation	Not federated, limited number of SNPs included, no evaluation	Data warehouse	None
LS-SNP (124)	Nonsynonymous SNPs, Protein sequences and models, pathways	Pipeline, links out to other sources	Data warehouse, links to outside sources	Case study
MutDB (125)	Missense SNPs	No evaluation, no federated integration system	Data warehouse, MySQL	None
PolyDoms (126)	Nonsynonymous SNPs	Lack of analysis tools, doesn't look at LD, no evaluation	Data warehouse, Oracle	None
PupaSuite (127)	Transcription factor binding, splicing, introns,	No federated data integration, no evaluation	Data warehouse	None

	exons, evolutionary, haplotypes			
SNPs3D (128)	Nonsynonymous SNPs and protein function, uses support vector machine learning	Not federated data integration,	MySQL database	Case study
SNPSelector (129)	Allele frequency, genotyping data	LD, dbSNP annotation, regulatory, repeat status	Data warehouse, MySQL	None

Table 3.1. Tables includes the comparisons of different previous SNP annotation systems.

3.4.2 Limitations of Current Approaches

However, one of the main limitations of these previous systems is that they all used a data warehouse approach when it comes to storing their data. This limits the usefulness and availability of the data since they are often out of date or not available. It also creates a challenge when integrating data across sources due to the need to have a common data model that is typically limited to what can be expressed using a relational data model. The exception to the warehouse approach is FastSNP which does use web wrappers, but doesn't include some kind of common data model which thus limits its extensibility. Furthermore, none of the previous systems performed a formal evaluation. They either did not evaluate their system, or they used a case study approach and looked at the results from a collection of SNPs. For example FastSNP, ranked 1,500 SNPs and used the fact that the majority of the

SNPs that were ranked highly had low minor allele frequency as validation, but this might be considered only a cursory form of functional evaluation. In addition the systems had relatively limited data sources. There is no one system that captures every data source related to SNP annotation.

Our system, SNP Integration Tool (SNPit) tries to improve upon all of these existing systems by implementing both a federated data integration system as well two different forms of inference for predicting SNP functionality. The reason we felt a general purpose data integration system was needed was because we wanted to create a system that had sufficient flexibility and a tiered architecture that would support iterative development of the SNPit system. The reason we adopted a federated model was we needed to ensure that the data source we linked to was up to date. The reason we felt different forms of inference were necessary was we felt that the users were more interested in a ranked list of SNPs. We also tried to incorporate a wide array of possible data sources into our system as well as an easy to use interface for our users. The reason we felt this was important was that having an easy to use interface with a comprehensive list of data sources that the system links to was an important requirement for the users. In addition, we formalized an evaluation of our system. We felt this was important because there had been no previous attempts at formal evaluations in previous SNP systems. All of these features help to differentiate our system from the existing systems.

3.5 Challenges to evaluation of SNP annotation systems/approaches

Though evaluation is important, there are numerous challenges when it comes to the evaluation of SNP annotation systems. The fundamental challenge is that there is currently no gold standard available when it comes to evaluating the predictive ability of any SNP annotation system. The reason for this is that there is no extensive catalogue of SNPs that have been proven to be causal. Genome wide association studies can provide a statistically significant correlation between the variant and the disease being studied. Additional evidence for a causal variant can be provided through wet-lab experiments, conducting replication studies using a different population, or by using the biological databases on the Internet. Even for the few studies that have replicated previously reported associations, such as the Crohn's disease (130), prostate cancer (131), and Wellcome Trust Consortium (36) articles, they only demonstrate the replication of previous studies and further illuminate the need for uncovering the biological mechanisms of the results.

Furthermore, even with statistical and biological evidence, one key component is still missing: the environmental component. Common diseases are the result of both genetic and environmental factors; right now there is no database that links together causal SNPs with environmental attributes. Furthermore, the

“common disease, common variant” hypothesis might not be correct (49). A recent survey of 560 SNPs from linkage studies, including 392 coding SNPs, reported that the majority of these SNPs which are thought to more likely have an impact on disease are not common SNPs (Figure 3.3). This would correspond to only 240,000–400,000 common cSNPs (out of a total estimated 11 million SNPs in the human genome) (132). This makes evaluating only common SNPs a challenge, as a training set of only common functional variants is not available and hard to extract out.

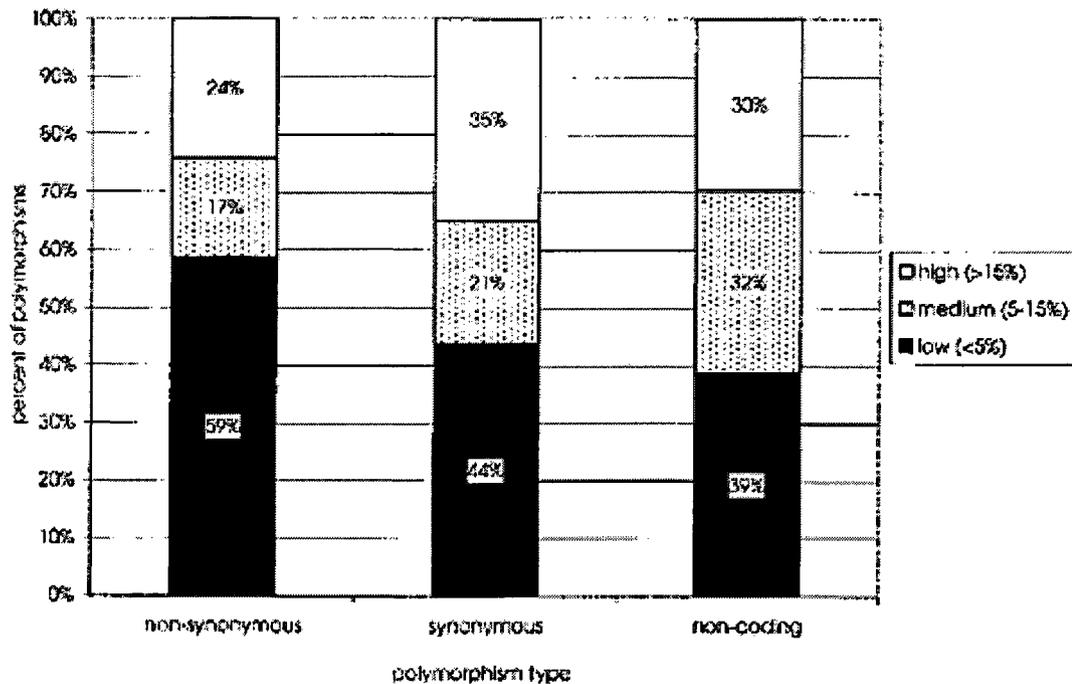


Figure 3.3. Recent survey findings that coding SNPs have lower minor allele frequencies than do silent SNPs (132).

3.6 Discussion

In this chapter, we examined how the different biological mechanisms factor into the predicted impact of a SNP and its ultimate functional annotation. Functional annotation is the first step to help determine utility and relevance of statistically significant association. We then looked at the different existing SNP annotation systems that use various techniques to try and capture these different biological mechanisms. Approaches to annotation (manual and automated/semi-automated) have limitations as summarized in section 3.3. Based on the gaps in current approaches we felt that there was a need to: a) integrate data across more diverse sources, b) to do so in a federated way to permit most current data to be used, c) to do so using a mediated schema to reconcile differences in modeling across sources, d) to do so in a modular way to permit easy addition of new sources, and e) a need to create a framework that supports both logical and probabilistic inference to permit pruning and ranking of the large result sets that come from this data integration approach. In particular we felt logical inference was important and likely to improve the rankings of functional SNPs because of previous successes in logical inference when applied to gene annotation. We also felt probabilistic inference was likely to be improve the rankings of functional SNPs because of previous successes in probabilistic inference when applied to gene annotation. Furthermore, none of the

existing systems use a formalized approach to evaluate the accuracy of their functional annotations and thus as discussed in Chapter 6 we developed and executed a formalized evaluation. The SNPit system implements a federated data integration approach as well as reasoning in the forms of logical and probabilistic inference in order to predict the likelihood that a SNP impacts the final phenotype. In the next two chapters, we will examine the elements of the SNPit system (Chapter 4), the details of the implementation of the SNPit system (Chapter 5), and in Chapter 6 present the approach taken to evaluation and the results of this evaluation.

Chapter 4: Foundations of the SNPit Data Integration System

4.1 Introduction

In the previous chapter, we discussed the current limitations of the existing SNP annotation systems. None of the existing systems use a federated data integration approach. None of the existing systems use reasoning in the forms of both logical and probabilistic inference. None of the systems are designed in a modular fashion to permit easy addition of new data sources or refinement of the mediated schema/common data model. In this chapter, background for implementation of our system, SNP Integration Tool (SNPit) will be presented. The key components of SNPit are a data integration platform, a logical inference module, probabilistic inference module, and a user interface.

4.2 Data Integration Concepts

To create an automated SNP system that can collect and analyze diverse SNP information, data integration was conducted on all the relevant sources of SNP functional information. Data integration is defined as the process of querying across heterogeneous data sources. Data integration is done in a transparent manner, allowing users to focus on the information they are interested in without having to deal with how the data is gathered (133). In addition, this avoids requiring the user

to be an expert in the individual biological sources (134). The user issues a single query and the system queries the individual data sources and integrates the results. A successful data integration system needs to be easy to query, easy to keep up to date, and easy to add new sources to. There are three aspects to data integration: how the data is stored, how the data is represented, and how the data is displayed.

4.2.1 Types of Data Integration- How the data is stored

How the data is stored depends on the integration architecture of the system, which can vary from data warehouses, database federations, data federations with mediated schema, and peer data management systems (135). A data warehouse consists of a single database with a central schema, with local copies of the diverse source data being stored. Advantages to data warehouses are faster response time for users to query the data as well as more control over the local data. Disadvantages to data warehouses include the fact that the data might not be up to date as well as the time and effort needed to maintain and expand the data warehouse (particularly the central data schema). Database federations, on the other hand, do not store local copies of the data sources; instead, they use a common data model and software code in the form of wrappers that connect to the different data sources. Advantages to database federations include the fact that the data is always up to date and the ease with which new data sources can be added. Disadvantages to database federations include the fact that querying of the data can be slow depending on how long it takes

to query each individual data source. Similar to a common data model, a mediated schema can be used in a federated data integration system to model all the different entities and relations represented in the separated data sources. In cases where the number of data sources makes it difficult to create one central mediated schema, a new idea that is currently being devised is the integration of separate mediated schemas, creating a peer data management system. (135)

All these types of data integration have their advantages and disadvantages, and which type of data integration is chosen depends on the goals of the application. For the SNPit system, we were dealing with a large number of data sources that needed to be integrated together as well as need to add new sources when necessary, we also needed to make sure that the data is always up to date. For these reasons, we decided to use a federated database with a mediated schema in a system that permits evolution of the schema relatively easily.

4.2.2 Types of Data Models – How the data is represented

How the data is represented depends on the type of data model used. The most common and widely used data model is the relational data model, where tables, rows, and columns are used to capture the data. Though well used and understood, relational data models can make the modeling of imprecise and uncertain data relationships difficult. Another approach to data modeling, semi-structured data modeling, allows for more fluidity. Semi-structured data modeling is represented as

labels and values, and can be graphed as nodes and edges. The most common format for semi-structured data modeling is XML. The data model that allows for the most complex representation of data and relationships is the ontology, which is a representation of the entities and relationships in whatever application area is being modeled. Ontologies have the ability to model complex objects, object classes, relationships, and functions (136). Due to the expressive nature of the SNP data sources, we decided to use an ontology (specifically the Protégé frames based knowledge representation system (137)) when building the mediated schema in our SNPit system. (135)

4.2.3 Types of User Interfaces – How the data is displayed

How the data is displayed influences how the user interacts with the system, this interaction is termed the user interface of the system. A user interface is defined as the method in which a system displays information to a user as well as how the user controls the system (138), there are two kinds of data interfaces that are commonly used: graphical user interfaces and text based user interfaces. Graphical user interfaces display the output in a graphical format to the user's computer screen, the user provides input to the interface through their computer's input devices. Text based user interfaces display the output in a primarily textual format.

In addition, user interfaces can be deployed using either a local or a client-server software architecture. Local stand alone applications require that the application first be installed on the user's local machine. Client-server applications in the form of web based user interfaces allows the user to input and receive output through the Internet, so long as the user has a web browser program. As compared to the local, stand alone user interface, the user does not have to download any additional software or programs when using a web based user interface. Due to the fact that we wanted to allow users to access the SNPit system through the Internet, we decided to use a web based text-based user interface.

4.3 BioMediator Data Integration System

The BioMediator data integration system chosen as the core of the SNPit system was developed at the University of Washington by the Bio-DIAG research group (Biomedical Data Integration and Analysis Group). We chose to use BioMediator as the underlying system of our SNPit system for a variety of reasons. It uses: a federated data integration system, mediated schema, easy to use query method, XML as a syntactic standard, and a frames based ontology as its common data model. This made it a good fit for our SNPit system because our system needed to: be able to have access to up-to-date information, have a flexible data model whenever new sources needed to be added on, be user friendly for those who access the system, and incorporate the various forms of data in which it is retrieved into one

common format. The later was particularly important for adding layers of logical and/or probabilistic inference on top of the integrated data. These attributes made BioMediator a good fit for our domain of interest.

4.3.1 Overview

BioMediator is a federated data integration system, which as previously described in section 4.1.1, means that the system queries the different data sources live and doesn't require a large, local repository of the data. The BioMediator system also uses a common data model in the form of a mediated schema, which allows for more flexible data modeling and easier updating. The components of BioMediator have a modular architecture, designed to perform data integration over multiple structured and semi-structured biologic data sources in a flexible way.

(139)

4.3.2 Architecture

Through the use of BioMediator, researchers can send a query which is transferred through the user query interface. The query then passes to the source knowledge base through the query processor layer. The source knowledge base is the mediated schema. The mediated schema acts as an ontology of the domain of interest, it contains an outline of the common objects and mappings of the individual

data sources. The query is then passed to the metawrapper which translates the query semantically, so that whatever terms the user is used in the mediated schema is translated into the terms the data sources uses. Finally, wrappers connect to the data sources (Tier #1), retrieves the data, and translates the data from whatever form it originally is into XML. The XML retrieved data then goes back through the metawrapper (Tier #2) and mapped onto the mediated schema (Tier #3). The retrieved data then goes back through the query processor and is accessed through the user query interface by the user (Tier #4). (Figure 4.1). The BioMediator system is able to connect to and query the data sources through http servlets allowing access to the individual data sources through URL parameters. The original source can be in a variety of formats, such as HTML or ASCII text, the wrapper queries the data and returns the results in XML. The metawrapper then maps this returned XML set onto the mediated schema.

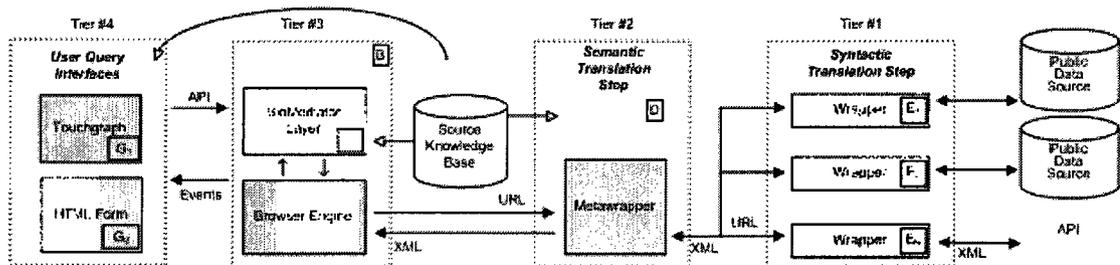


Figure 4.1. Overview diagram of BioMediator system (139).

4.3.3 Mediated Schema

The main component of the BioMediator system is the source knowledge base which includes both a source catalog as well as the mediated schema. The source catalog includes the descriptions of the concepts and relationships in the mediated schema. The mediated schema acts like a global schema, except that it only needs to model those concepts requested by the developer, making the mediated schema much more flexible than a global schema (140). The mediated schema includes a hierarchical listing of all the concepts and relationships in the domain of interest, it can be accessed through the Protégé Knowledge Base (137).

4.3.4 User Interfaces

The core BioMediator comes with a graphical user interface, implemented through a visualization software named Touchgraph (141). The result sets are displayed in a graph with the entities as nodes and the relationships as edges (Figure 4.2). However, some users find the graph based display uneasy to use, particularly once the result sets become very large. Due to this concern and to enable users the ability to interact with the SNPit system without having to download any additional software, we created a text based interface for the SNPit system. The Touchgraph user interface uses a local software architecture while the text based user interface uses a client-server software architecture, with the client application running in a web browser.

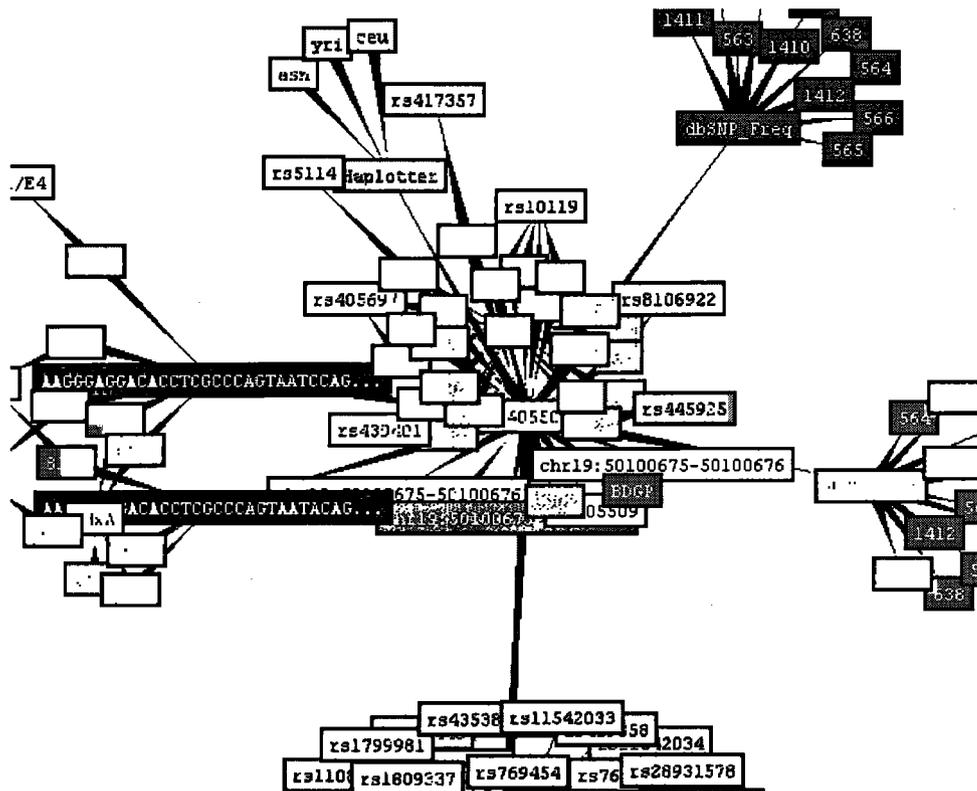


Figure 4.2. Graphical based interface implemented using the Touchgraph software.

4.4 Data Integration and Logical Inference

Previous work has been conducted on how logical inference in the forms of expert rules could be applied to the BioMediator system and result graphs. Due to the fact that BioMediator is highly extensible, a rules reasoning system plug-in named Java Expert System Shell (Jess) (142) was incorporated into the BioMediator system. (143)

4.4.1 Overview

Logical inference in the form of expert rules systems try to capture the knowledge that experts can bring to a particular domain of interest. This expert knowledge is captured in a computer program through the use of rules. Expert rules systems are made up of two main components: 1) working memory, where facts concerning the domain of interest are stored, and 2) rule memory (also called production rules), which are the facts created to capture the expert knowledge (144). Rules that are created in the rule memory can be used to add to or edit the rules in working memory. Thus, new facts can be created from pre-existing knowledge through the use of expert rules.

4.4.2 Jess Logical Inference Engine

The facts and rules that can be created in the Jess inference engine are built off of the concepts of propositional and predicate logic. A proposition is defined as a fact that can be either true or false, any proposition can directly be translated into a Jess fact. Jess rules, which can be thought of as a form of predicate logic, can then be used to answer questions about the propositions. Predicates are defined as a

description of the object, when they are instantiated with actual objects, they form simple propositions (145). Rules in Jess are made up of simple IF-THEN statements, when the rule is executed, a fact based on the THEN part of the statement is added or removed from the working memory. (144)

There are many advantages to using a rules based inference engine. Rules can stand alone even if they are never executed, thus, they have a modular structure, and new rules can easily be added or deleted. Furthermore, rules and facts are inherently easier to understand from a semantic standpoint and they also are directly manifested from the knowledge of experts in a domain of interest.

4.4.3 Prior Work Into Logical Inference and BioMediator

Prior work into logical inference building upon both BioMediator and the Jess rules inference engine was conducted by Eithon Cadag, who designed and evaluated another protein annotation system using anonymous sequences as part of his M.S. thesis. Previous logical inference systems that dealt with gene annotation were limited by a pipeline approach to data integration and a lack of flexibility in inference capabilities. This work concluded that a gene annotation system build upon rules and facts in a rules based expert system performed slightly better than a similar annotation system and similarly to human annotation (the de facto gold standard). This work also found that the evaluation results could have been

improved for rules based inference of protein sequences if the rules were modified to account for similar annotation names. (143, 146).

4.5 Data Integration and Probabilistic Inference

There is an uncertainty present at various levels in biomedical data. This is particularly true in the SNP functional annotation arena. This uncertainty in the data translates into inherent uncertainties in the data sources that integration systems connect to. To try and capture this uncertainty, the NSF-Funded UII project (UII: Uncertainty in Information Integration) (NSF Grant: NSF IIS-0513877) focused on developing formal models of this uncertainty and implementing this model in the form of a functional system (147)

4.5.1 Overview

The Uncertainty in Information Integration (UII) project builds upon the BioMediator project by incorporating modules that model and incorporate uncertainty metrics (also called uncertainty parameters) into the core BioMediator system. These uncertainty metrics are probabilistic values ranging from 0 to 1.0 which represent the amount of uncertainty in a data source or in how the data sources interlink to one another. These uncertainty metrics are applied to the mediated

schema both before and after any of the entities or relationships have been queried and instances for the objects are created. The uncertainty metrics are then automatically combined into a total score called a relevance score, which represents the amount of certainty the user has in the result set returned when an initial query is sent. The following section, 4.3.2, describes the details of the uncertainty model and metrics and the determination of the relevance score. (147)

4.5.2 Uncertainty: The Uncertainty In Information Integration (UII) System

The uncertainty measures in the UII system come from four sources: 1) the amount of trust one has in the individual data source, 2) the amount of trust one has in the linkages between data sources, 3) the amount of trust one has in an actual data record, and 4) the amount of trust one has in an actual linkage between actual data records. These metrics are captured in four different measures that are applied to the mediated schema and to the result graph (Table 4.1).

Uncertainty measure	Measure description
Ps	Prior belief in the quality of an individual data source

Qs	Prior belief in the quality of the relationship between two data sources
Pr	Post query belief in the quality of the individual data record, calculated after individual the query is submitted
Qr	Post query belief in the quality of the relationship between two individual data records, calculated after the initial query is submitted

Table 4.1. Uncertainty metrics for four different measures.

These uncertainty metrics are specific to the individual data sources and data records, they do not necessary translate across different data sources and data records. In order to come up with one summarized belief score, the members of the UII project decided to build upon network reliability theory, a model that helps to calculate the connections in a graph (148). The final UII score is the probability that a particular node in question can be reached from the initial seed query node. Each node as it progresses along the graph is summarized by multiplying the Ps value with the Pr value. Similarly, each edge along the graph is summarized by multiplying the Qs value with the Qr value.

Both the Monte Carlo methods and network reliability theory is used during the calculation of the UII score (also referred to as the relevance score). The UII score measures the chances that two nodes in a network are connected. For example, Figure 4.3 shows a minimum network consisting of one query (Q) node and one result (R) node, network reliability theory tries to figure out the chances that the R node is connected to the Q node.



Figure 4.3. Basic connection between a Query (Q) and a Result (R).

For more complicated networks, though, the calculation becomes more difficult the farther the Result node is from the Query node. If it was only on a local level, the probability (Pr) of node 4 (N_4), for example, would be the probability of the source of node 4 (Ps_4) times the probability of the record of node 4 (Pr_4). Similarly, the probability (Pr) of edge4 (e_4), would be the probability of the source of edge 4 (Qs_4) times the probability of the record of edge 4 (Qr_4) (Figure 4.4).

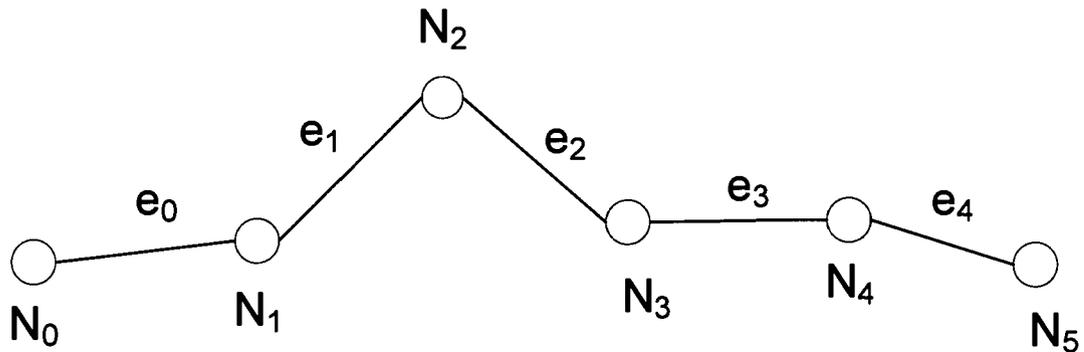


Figure 4.4. Extended network showing nodes(N) and edges(e).

This concept of measuring the connection between the query node and the result node, though, becomes extremely complicated and untractable once you start adding on multiple paths. In order to solve this problem, a Monte Carlo simulation is performed. Each of the nodes and edges have a trial vector (a list of numbers) applied to them and the nodes have a success vector applied to them.

The trial vector has N bits (0 or 1) depending on the number of trials (N). A 0 or 1 is assigned based on a random draw, based on their associated probabilities. A success vector of N bits is also assigned to each one of the nodes. A depth first search is then conducted and using the logical AND, the head node success vector, the edge trial vector, and the tail node trail vector is all added together to create a new vector. Finally, that new vector is combined with the current success vector using logical OR to arrive at the final success vector. To arrive at the UII score or sometimes referred to as the relevance score, we assign the letter k to the number of

1 bits in the final success node, then we divide that by the number of trials, N . The final score becomes k/N . (149)

4.5.3 Prior Work Into Probabilistic Inference and BioMediator

Prior work into probabilistic inference building upon both BioMediator and the UII project was conducted by Brent Louie, who designed and evaluated a protein annotation system named BioMiner using the UII extensions to BioMediator. Previous probabilistic inference systems in the biological domain was limited to specific models that were static and did not incorporate data integration and uncertainty metrics. This work concluded that a protein annotation system built upon the uncertainty metrics and relevance scores as described previously performed better than other protein annotation systems. This work also found that the uncertainty measures were fairly robust in terms of how they performed in the final protein annotation system with system performance stable over a range of uncertainty parameters chosen. (150, 151)

4.6 Discussion

It has been previously demonstrated that BioMediator is an effective federated data integration system, one that is extensible and serves as a great platform for including logical and probabilistic inference. In previous work, both logical and probabilistic inference performed well independently of one another

compared to similar annotation systems in the domain of protein annotation. However, so far, no one has combined both probabilistic and logical inference in the context of a general purpose data integration system. Furthermore, no one has looked at how the combination of federated data integration, probabilistic inference, logical inference, and probabilistic and logical inference combined performs in compared to each other any domain. Finally no one has looked at the application of a general data integration system to the domain of SNP annotation. We built the SNPit tool to explore these questions in the context of determining the feasibility and value of federated data integration combining logical and probabilistic inference for SNP annotation. In the next chapter (Chapter 5) we describe how the SNPit system was built using the components and prior work described in Chapter 4.

Chapter 5: Determining Feasibility via Implementation of the SNPit System

5.1 Introduction

In the last chapter, we described the basic components behind the BioMediator system. We also introduced the concepts of probabilistic and logical inference. In Chapter 5, we describe the implementation of the SNP Integration Tool (SNPit), building upon the aforementioned BioMediator federated data integration system as well as prior work in the areas of both probabilistic and logical inference. After describing in detail the base SNPit system we describe the evolution of the system to incorporate various combinations of inference.

5.2 Design of the SNPit System

The baseline federated data integration system is built on top of the BioMediator system, from there additional inference in the either probabilistic or logical forms were laid on top of the baseline federated data integration system by building plug-ins in the form of Jess or the UII algorithm. Ultimately both logical and probabilistic inference were combined into a single version of SNPit.

5.2.1 Building upon BioMediator Core System

The core SNPit system (see also section 5.3) can be divided into three parts: the sources that SNPit links to, the BioMediator federated integration system that SNPit sits on top of, and the two different data interfaces that users can access the SNPit through (Figure 5.1). A high level overview of the function of SNPit is as follows. The biologist begins by querying the system for information about a particular entity (e.g. SNP, gene, protein, phenotype) using one of two methods, either using a graph-based GUI application or using a servlet-based HTML form. The user query is passed to the query processor which provides an API for launching and managing queries posed against the common data model. From there, a translational step is accomplished through the metawrapper, which translate the queries from the common data model into source specific queries. From there, wrappers act as an interface to pass the remapped queries through to the data sources. Data sources return results in native format, which are translated to XML syntax by the wrappers. The metawrapper then applies mapping rules in translating the XML result streams to the common data model semantics. The query processor then retrieves that XML data from the metawrapper, organizes it and generates events which can be used to synthesize a navigable, graph-based representation of the result set. The results can then be interfaced through either the graphical user interface or the text-based web servlet. In later versions of SNPit (5.4.2 and 5.4.3 and 5.4.4) logical and probabilistic inference alone and in combination are performed across the result graph.

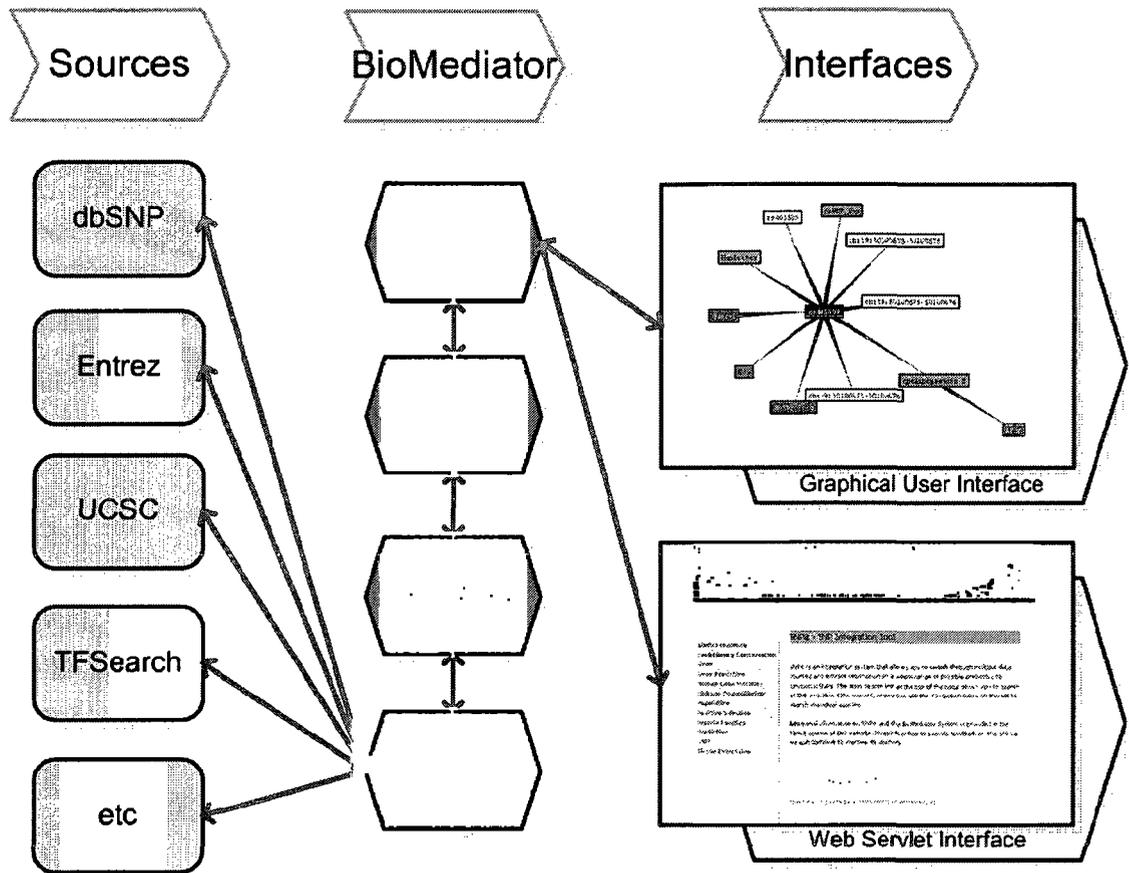


Figure 5.1. Diagram of the SNPit system (152)

5.2.2 Building upon Prior Logical Inference Extensions to

BioMediator

As previously mentioned in Chapter 4, section 4.4, logical inference had been previously implemented as a plug-in within the BioMediator federated data integration system. We extended the logical inference component of BioMediator in

order to model the expert knowledge available for the purpose of SNP annotation. Section 5.3.4 goes into details on the implementation of logical inference into the SNPit system. Both the logical and probabilistic extensions for SNPit were designed to be interoperable with each other to permit both independent and combined logical/probabilistic inference.

5.2.3 Building upon Prior Probabilistic Inference Extensions to BioMediator

As previously mentioned in Chapter 4, section 4.3, probabilistic inference had been previously implemented as the UII project for the BioMediator federated data integration system. We extended the UII project so that we could measure the uncertainty in the entities and relationships related to SNP annotation. Section 5.3.3 goes into details on the implementation of probabilistic inference into the SNPit system.

5.3 Implementation of the core SNPit System

In this section, we will describe in detail how the core SNPit system was implemented by building upon BioMediator and using inference techniques such as logical and probabilistic inference with the goal of creating an easy to use, all

purpose, modular, extensible SNP annotation system that can be used by researchers in many different scenarios. One scenario might be after a researcher has conducted the first stage of a genome wide association study; after the initial statistically significant SNPs have been revealed, the researcher might want to narrow down the list of possible SNPs in order to conduct a second stage of genotyping. Another scenario might be one in which the researcher is taking a candidate gene approach and wants to understand the biological background of the SNPs they have found to be associated with a particular disease in journal articles. There are a lot of different scenarios in which a tool that integrates and provides some kind of inference on SNP annotation might be useful. SNPit tries to provide such a tool through links to data sources, a common data model that attempts to model the SNP annotation, inference from both a logical and probabilistic standpoint, and an interface that is accessible over the Internet.

5.3.1 Data Sources

The SNPit core system integrates data from eleven main data sources: dbSNP, EntrezGene, HGMD, Haplotter, GVS, SIFT, UCSC Phastcons, UCSC Genscan, UCSC GNFAtlas2, and TFSearch. Table 5.1 lists the main sources and their descriptions. These sources were chosen because they all contribute information to potential biological outcome of a SNP. A description of each data source follows.

dbSNP	central repository database for both single base nucleotide substitutions and short deletion and insertion polymorphisms (47)
EntrezGene	gene-centric database that includes information on nomenclature, chromosomal localization, gene products and their attributes (153)
HGMD	provides publication evidence to genes responsible for human inherited diseases (154)
Haplotter	web tool that includes evidence for positive selection within the human genome, two statistical measures that look at the linkage disequilibrium of positively selected alleles and frequencies of polymorphisms are provided (155)
GVS	local database that provides access to information in dbSNP and includes tag SNP and linkage disequilibrium analysis (156)
SIFT	resource for predicting the functional impact of nonsynonymous coding SNPs, algorithm sorts tolerant from intolerant polymorphisms (157)
UCSC Phastcons	predicts evolutionary conservation of noncoding SNPs based on aligning genomic sequences of different species (158)
UCSC Genscan	predicts whether the SNP lies in a region that is likely to be gene region, based on transcriptional, translational, and donor/acceptor splicing signals (159)
UCSC GNFAtlas2	displays gene-centric expression of polymorphisms over 79 human tissues (160)
TRANSFAC	Database with transcription factor binding sites(161)
BDGP	An analysis tool for predicting splice sites (162)

Table 5.1. List of data sources that SNPit links to, along with brief descriptions of each source.

The main repository for SNP data is dbSNP, currently there are over 7,000,000 human SNPs entered for the latest build of dbSNP (build 130). dbSNP has a variety of different search options and results. Users can submit a SNP through their submission process, SNPs are first assigned a unique submitted SNP id (SS#), then the scientists at NCBI map all the SS#s onto a genomic contig (contiguous part of the genome) and assigns them by cluster called a “reference SNP cluster”, or

“refSNP”, each cluster is then assigned a unique RefSNP ID number (rs#). (163) dbSNP provides the allele frequencies and functional analysis piece of the puzzle when it comes to SNP annotation.

EntrezGene is another site sponsored by NCBI, it contains a variety of gene related information including both the nucleotide and amino acid sequences of a gene as well as gene name, gene id, and predicted gene annotation. EntrezGene is used primarily as a source for identifying which SNPs fall within a known gene segment.

The Human Gene Mutation Database (HGMD) is an online database that provides information on germline mutations for over 85,000 lesions (injuries to the gene) in 3,253 genes (164). The resource attempts to provide information on both different forms of disease causing mutations as well as potentially functional polymorphisms. Mutations and polymorphisms information is supported by published literature which the developers of HGMD find through both manual and computational methods. SNP information is available for three categories: missense or nonsense, splicing, and regulatory. SNPit utilizes HGMD as its main source for evaluation purposes.

The Haplotter website tries to capture the notion of positive selection, which occurs when a particular SNP is selected for due its evolutionary benefits, this could be due to a variety of environmental or lifestyle reasons (155). When there is no evolutionary forces at play, it usually takes a long time for newer polymorphisms to

reach a high level of allele frequency; furthermore, the linkage disequilibrium around that SNP will decay due to recombination (165). The Haplotter data source proposes a new statistic which they entitle the iHS (integrated haplotype score) to try and capture this kind of positive selection. Results are available in three different ethnic groups: Japanese and Han Chinese East Asian population (ASN), northern and western European origin (CEU), and Ibadan, Nigeria Yoruba (YRI) individuals. SNPs with iHS scores that are extreme in both the positive and negative direction are thought to be interesting. (155)

The Genome Variation Server (GVS) is a database available through the University of Washington's SeattleSNPs Program for Genomic Applications (PGA), the resource supplies information on linkage disequilibrium and tagSNPs. Linkage disequilibrium is calculated using Pearson's correlation coefficient (166):

$$r^2 = D^2 / p_{A1}(1-p_{A1})p_{B1}(1-p_{B1}) .$$

The Sorting Tolerant from Intolerant (SIFT) is a database available through the Fred Hutchinson Cancer Research Center. The program uses sequence homology to predict which polymorphisms cause a change in the amino acid sequence that ultimately impacts the protein. The SIFT program uses the concept that protein families tend to be conserved along the genome, if the polymorphism being submitted changes the amino acid to one of a different set of properties, then it is predicted to be deleterious (167). Scores of less than 0.05 are predicted as intolerant.

The Phastcons track in the UCSC genome browser uses the phylogenetic hidden Markov model (phylo-HMM) statistical model and looks across the genome of vertebrate, insect, worm, and yeast genomes and two different states (168). The scores range from 0 to 1, with higher scores being an indication that the region is more evolutionarily conserved, and thus a SNP at one of these locations would potentially have more of an impact on the protein product.

The Genscan track in the UCSC Genome Browser is a gene identification program that integrates information on exons/introns, splicing signals, and similarity to known protein sequences to predict if the polymorphism is within a predicted gene region (169).

The GNF Gene Expression Atlas 2 track in the UCSC Genome Browser includes information collected from an array experiment of 79 human tissues (170). The experiment scores range from -4 to 4, a zero or negative value indicates that there was low expression in that tissue.

TFSearch is an online database that provides transcription factor binding site information via both cis-acting and trans-acting (segments of the DNA that regulate genes either in the same region or in a different region via transcription factor mechanisms. There is, unfortunately, sparse literature on the details of the implementation of this tool (171).

Finally, BDGP is an analysis tool that uses a back propagation feed forward neural network with one hidden layer, along with a 43 human gene training set to

predict both the 5' and 3' binding sites for eukaryotes (172). The scores range from 0 to 1, with a higher score being an indication of a likely splice site.

These data sources are interfaced with the core SNPit system via the wrappers in BioMediator. The code for the wrappers to these sources are in Appendix A. Once the information is retrieved, the wrappers transform the data from its original format into XML. From there, the metawrapper maps the returned XML data onto the common data model through various mapping directives. The common data model, or also called the mediated schema, can be thought of as a model of the SNP annotation world.

5.3.2 Common Data Model for SNPit

The common data model (mediated schema) of the SNPit system is an attempt to model the different entities and relationships related to SNP annotation across diverse sources of information. Different entities and relationships are linked together in either one directional or bi-directional arches (pointers). Relationships can also be one-to-one or one-to-many. Figure 5.2 is a diagram of the overall scheme of the mediated schema.

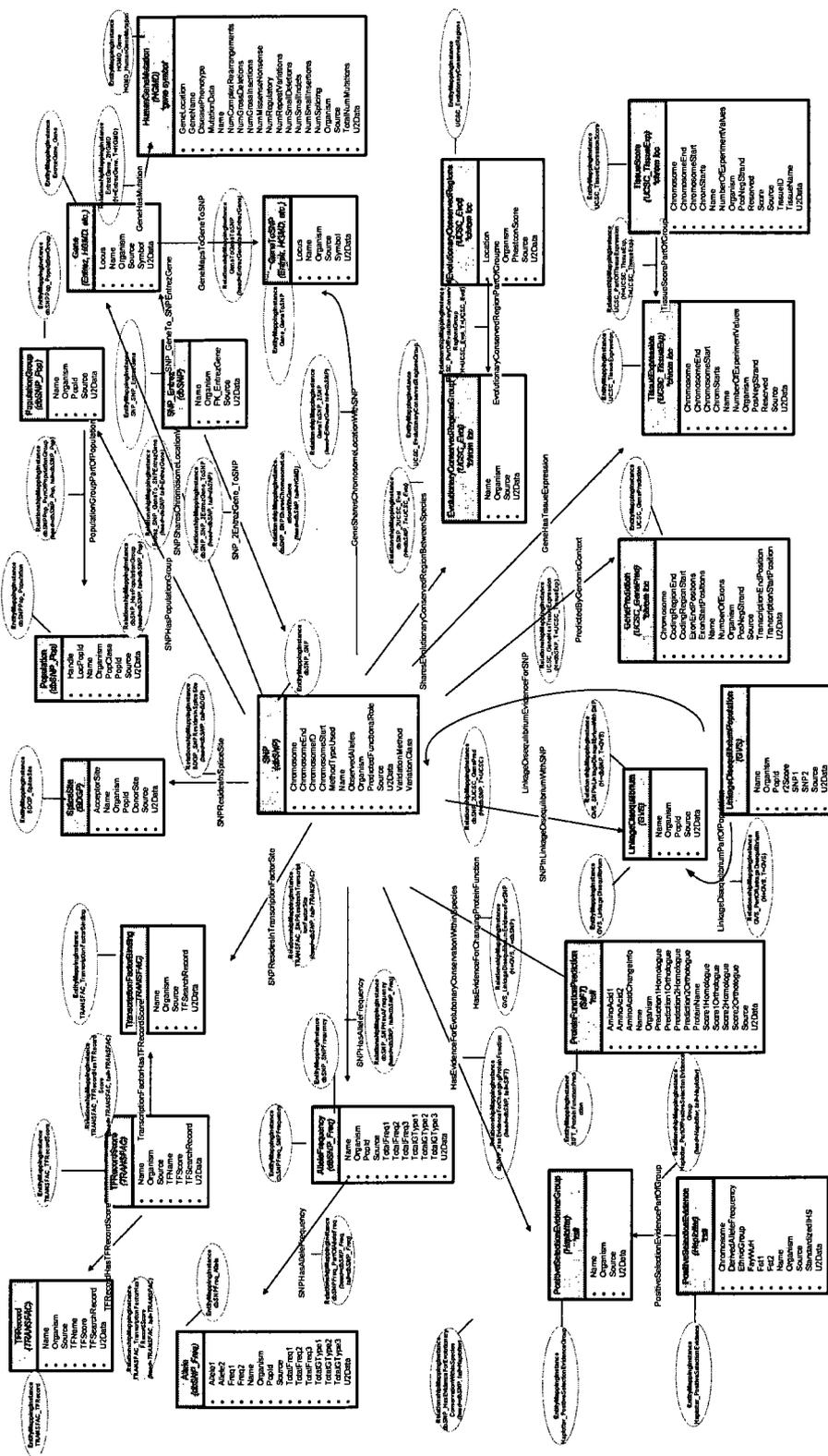


Figure 5.2. Mediated schema of SNPit (152).

If we examine the mediated schema in detail, we can see the entities, relationships, and databases implemented in SNPit. Table 5.2 is a description of each of the entities and their forward pointers.

The SNP entity, for example, has a pointer going forward to the entities: Splice Site, Transcription Factor Binding, Allele Frequency, Positive Selection Evidence Group, Linkage Disequilibrium, Gene Prediction, Tissue Expression, Evolutionary Conserved Regions Group, Gene To SNP, Population Group, and Gene (Figure 5.2).

Database	Entity	Forward Pointer
dbSNP	SNP, PopulationGroup, AlleleFrequency	SpliceSite, PopulationGroup, Gene, GeneToSNP, EvolutionaryConservedRegionsGroup, TissueExpression, GenePrediction, LinkageDisequilibrium, ProteinFunctionPrediction, PositiveSelectionEvidenceGroup, TranscriptionFactorBinding, Population, Allele
EntrezGene	Gene	GeneToSNP, HumanGeneMutation
HGMD	HumanGeneMutation	
BDGP	SpliceSite	
UCSC	EvolutionaryConservedRegionsGroup, EvolutionaryConservedRegions, TissueExpression, TissueScore, GenePrediction	EvolutionaryConservedRegionsGroup TissueExpression
GVS	LinkageDisequilibrium, LinkageDisequilibriumPopulation	LinkageDisequilibrium
SIFT	ProteinFunctionPrediction	
Haplotter	PositiveSelectionEvidenceGroup, PositiveSelectionEvidence	PositiveSelectionEvidenceGroup
TRANSFAC	TranscriptionFactorBinding, TFRecordScore	TFRecord

Table 5.2. Database and entities for the common data model.

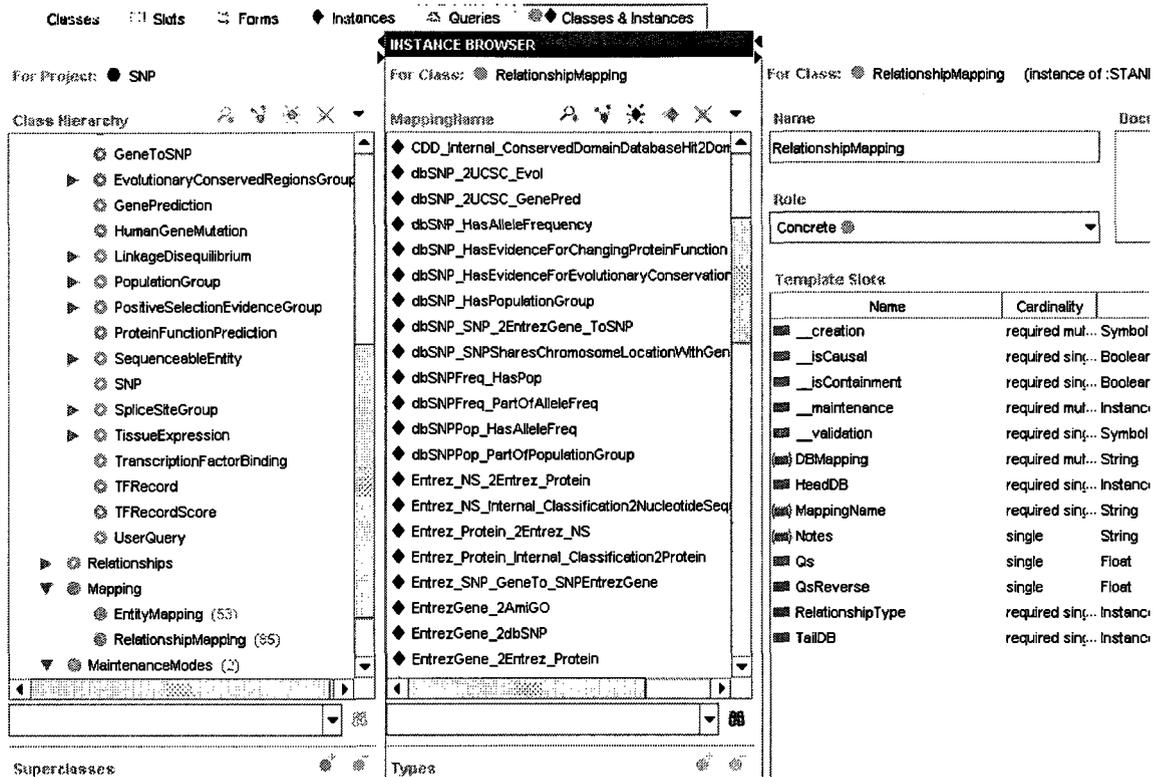


Figure 5.3. The common data model is implemented via the Protégé Knowledge Base.

As mentioned in section 4.3.3 of the last chapter, the common data model is implemented through the Protégé Knowledge Base (137). The Protégé Knowledge Base is an ontology editor and allows the common data model to be adjusted and altered as necessary. Modification of the SNP mediated schema involves creations of different class and instance frames. The attributes of each the classes can be

added as a slot to the frame (Figure 5.3). The full Protégé data model is available in Appendix B of this dissertation.

5.3.3 Logical Inference for SNPit

Logical inference rules were included in the SNPit system by creating a decision tree using the biological principles which were previously mentioned in Chapter 3, section 3.1. Previous literature which summarize the biological properties on which SNPs can be prioritized based on functional predictability was also considered in the creation of the decision tree (114, 173) (Figure 5.4). The classification of the polymorphism, whether it is a nonsynonymous or synonymous SNP, for instance, would be helpful towards indentifying the SNP's functional annotation. The decision tree was revised as appropriate through the input of two experts. Weighted scores were then assigned to the nodes of the decision tree; these scores were on a 1 to 4 scale, with higher scores indicating a stronger likelihood of predictive potential.

Table 1 | Priorities for single-nucleotide-polymorphism selection

Type of variant	Location	Functional effect	Frequency in genome	Predicted relative risk of phenotype
Nonsense	Coding sequence	Premature termination of amino-acid sequence	Very low	Very high
Missense/non-synonymous (non-conservative)	Coding sequence	Changes an amino acid in protein to one with different properties	Low	Moderate to very high, depending on location
Missense/non-synonymous (conservative)	Coding sequence	Changes an amino acid in protein to one with similar properties	Low	Low to very high, depending on location
Insertions/deletions (frameshift)	Coding sequence	Changes the frame of the protein-coding region, usually with very negative consequences for the protein	Low	Very high, depending on location
Insertions/deletions (in frame)	Coding or non-coding	Changes amino-acid sequence	Low	Low to very high
Sense/synonymous	Coding sequence	Does not change the amino acid in the protein — but can alter splicing	Medium	Low to high
Promoter/regulatory region	Promoter, 5' UTR, 3' UTR	Does not change the amino acid, but can affect the level, location or timing of gene expression	Low to medium	Low to high
Splice site/intron-exon boundary	Within 10 bp of the exon	Might change the splicing pattern or efficiency of introns	Low	Low to high
Intronic	Deep within introns	No known function, but might affect expression or miRNA stability	Medium	Very low
Intergenic	Non-coding regions between genes	No known function, but might affect expression through enhancer or other mechanisms	High	Very low

UTR, untranslated region.

Figure 5.4. Literature summary of the various methods in which the location on which a SNP resides impacts the potential function of the SNP (173).

Once the heuristic weights are assigned, each path's final node in the decision tree is calculated by multiplying all the previous node's weighted values together. The decision tree (Figure 5.5) has heuristic weights assigned to the different nodes along its branches. For example, SNPs that are in the coding region and are also nonsynonymous and damaging would have the highest predicted value and a weight of 3.375 was assigned to that branch of the tree. On the other hand, if the SNP is non-coding, is not in a UTR region, is only in a flanking region, and has no predicted transcription binding sites associated with it, then a lower score of 0.5 would be assigned to its branch of the tree. Jess rules then used the information gathered from the BioMediator federated integration system along with the logical inference described in the decision tree to set up a collection of rules for the purposes of SNP annotation.

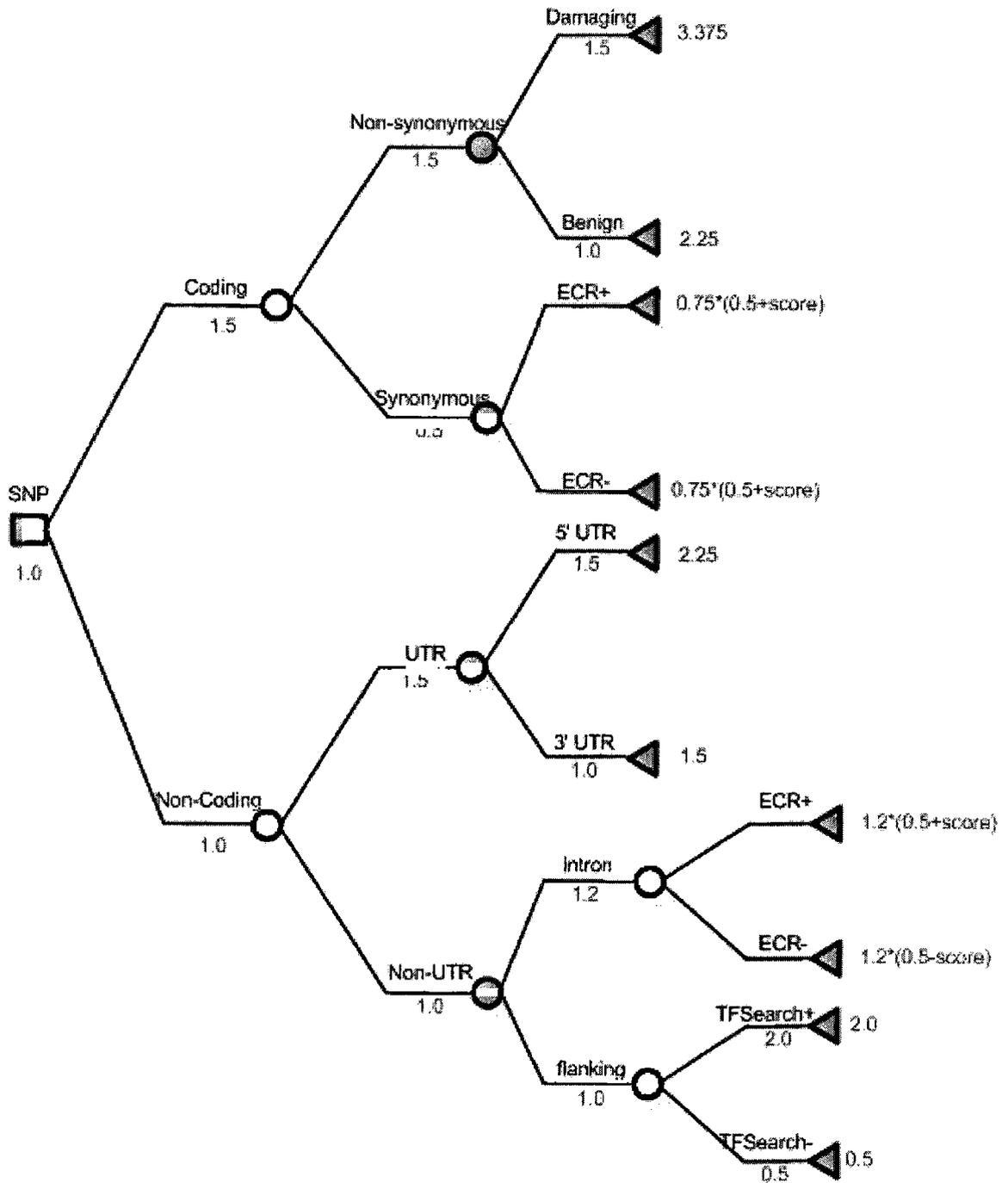


Figure 5.5. Decision tree with heuristic weights assigned to the branches.

5.3.4 Probabilistic Inference for SNPit

We assigned probabilistic measurements for the Pr, Qr, Ps, Qs to the different sources related to SNP annotation. As previously mentioned in Chapter 4, section 4.3, Pr is the probability that we trust an individual record; Qr is that we trust we have in the link between two records; Ps is the probability that we trust the data source that the record comes from in general, and Qs is the probability that we trust the link between data sources. Table 5.3, for example, shows the actual probability values that were assigned for the Ps values. The Pr values were assigned through the Belief Resolver java files (see also Appendix C – Belief Resolver java files). They were assigned Pr scores based on what information each of the records returned. For example, for each of the records in dbSNP, we were able to assign a Pr score based on their validation status, for SNPs that were genotyped through the HapMap project were assigned a higher value than those that were submitted by multiple laboratories.

Source	Entity	Ps (a priori belief in how good a node is)	Rationale	Pr (post belief after the record is examined)	Rationale
dbSNP	SNP	0.9	<ul style="list-style-type: none"> Well known source, but not necessarily well validated Not everything is reliable 	depends	<ul style="list-style-type: none"> dbSNP includes information on validation status, (validated by multiple, independent submissions to the refSNP cluster, validated

			<ul style="list-style-type: none"> • Less than half have allele frequency data 		<p>by frequency or genotype data, validated by submitter confirmation, all alleles have been observed in at least two chromosomes apiece, genotyped by HapMap project)</p> <ul style="list-style-type: none"> • submitter number maybe
dbSNP	Allele	0.9	<ul style="list-style-type: none"> • Pollution of data is possible 	depends	
dbSNP	Allele Frequency	0.9		depends	
dbSNP	Population	0.9		depends	
dbSNP	Population Group	0.9		depends	
EntrezGene	Gene	0.95	<ul style="list-style-type: none"> • Links • Not all genes are annotated 		
TRANSFAC	Transcription Factor Binding	0.7	<ul style="list-style-type: none"> • Poorly characterized transcription factors (incomplete data sets) 		
TRANSFAC	TFRRecord	0.7			
BDGP	Splice Site	0.8	<ul style="list-style-type: none"> • More reliable training set 		
HGMD	Human Gene Mutation	0.8	<ul style="list-style-type: none"> • Literature may be wrong 		
Haplotter	Positive Selection Evidence	0.8	<ul style="list-style-type: none"> • Pieces that aren't well characterized 		
SIFT	Protein Function Prediction	0.85	<ul style="list-style-type: none"> • More specific knowledge of the region (protein) 		
GVS	Linkage Disequilibrium	0.95	<ul style="list-style-type: none"> • Less error in measurement 		
UCSC	Gene Prediction	0.8			
UCSC	Tissue Expression	0.85	<ul style="list-style-type: none"> • Directly measured 		
UCSC	Tissue Expression Score	0.8			
UCSC	Evolutionary Conservation	0.8			
UCSC	Evolutionary Conservation Score				

Table 5.3. Descriptions of Ps and Pr values and the rationale for who the values were selected.

Similarly, Qs and Qr scores were assigned based on the belief we had in the trustworthiness of the linkages between sources (Table 5.4). Sources that are linked together via their rs# or chromosome locations were assigned a higher Qs value because their linkages were deemed to be more trustworthy than secondary linkages through sequence or gene names.

Relationship	Sources Involved	Qs(a priori belief in the link between sources based on experts and keys)	• Rationale	Qr (post belief in specific edge record)	Rationale
TFRecordHasTFRecordScore	TRANSFAC	1.0	• Direct linkage	depends	
TranscriptionFactorHasTFRecordScore	TRANSFAC	1.0	• Direct linkage	depends	
SNPHasAlleleFrequency	dbSNP_Freq	1.0	• Direct linkage	depends	
SNPResidesInTranscriptionFactorSite	dbSNP to TRANSFAC	0.9	• Linked by sequence	depends	
SNPHasAlleleFrequency	dbSNP to dbSNP_Freq	1.0	• Direct linkage	depends	
SNPResidesInSpliceSite	dbSNP to BDGP	0.9	• Linked by sequence		
SNPHasPopulationGroup	dbSNP to dbSNP_Pop	1.0	• Direct linkage		
PopulationGroupPartOfPopulation	dbSNP_Pop	1.0	• Direct linkage		

SNP_2EntrezGene_ToSNP	dbSNP_Pop				
SNP_GeneTo_SNPEntrezGene	head=dbSNP tail=EntrezGene	0.9	• rs number link		
GeneHasMutation	H=EntrezGene, T=HGMD	0.8	• Gene name link		
SharesEvolutionaryConservedRegionBetweenSpecies	H=dbSNP, T=UCSC_Evo	0.9	• Chromosome location link • Might have version problems; if it's the same build, it's higher than the sequence link, it not than lower		
EvolutionaryConservedRegionPartOfGroup	UCSC_Evol	1.0	• Direct link		
GeneHasTissueExpression	dbSNP to UCSC_Tissue Exp	0.9	• Chromosome location link		
TissueScorePartOfGroup	UCSC_Tissue Exp	1.0	• Direct link		
PredictedByGenomicContext	H=dbSNP, T=UCSC_GenePred	0.9	• Chromosome location link		
SNPInLinkageDisequilibriumWithSNP	dbSNP to GVS	0.9	• rs number link		
LinkageDisequilibriumPartOfPopulation	GVS				
HasEvidenceForChangingProteinFunction	dbSNP to SIFT	0.9	• rs number link		
HasEvidenceForEvolutionaryConservationWithinSpecies	dbSNP to Haplotter	0.9	• rs number link		
PositiveSelectionEvidencePartOfGroup	Haplotter	0.9	• rs number link		

Table 5.4. Descriptions of Q_s and Q_r values and the rationale for who the values were selected.

5.3.5 User Interfaces

SNPit was implemented with two user interfaces: a graphical user interface as well as a text-based interface. As previously mentioned in Chapter 4, section 4.2, the graphical user interface was implemented via Touchgraph. SNPit's graphical user interface (GUI) needs to be downloaded to the user's local computer in order to be accessed. As shown in Figure 5.6, the user submits an initial seed, in this example, rs405509 (rs405509 is an ID in dbSNP, previously described in section 5.3.1) and the GUI expands out to all the different sources of information on that initial query. The various data sources return the information in the form of nodes and edges. In the rs405509 example, linkage disequilibrium information for rs405509 is displayed, the results indicate that rs405509 is correlated with rs10119, rs8106922, rs225925, rs439401, and rs405697.

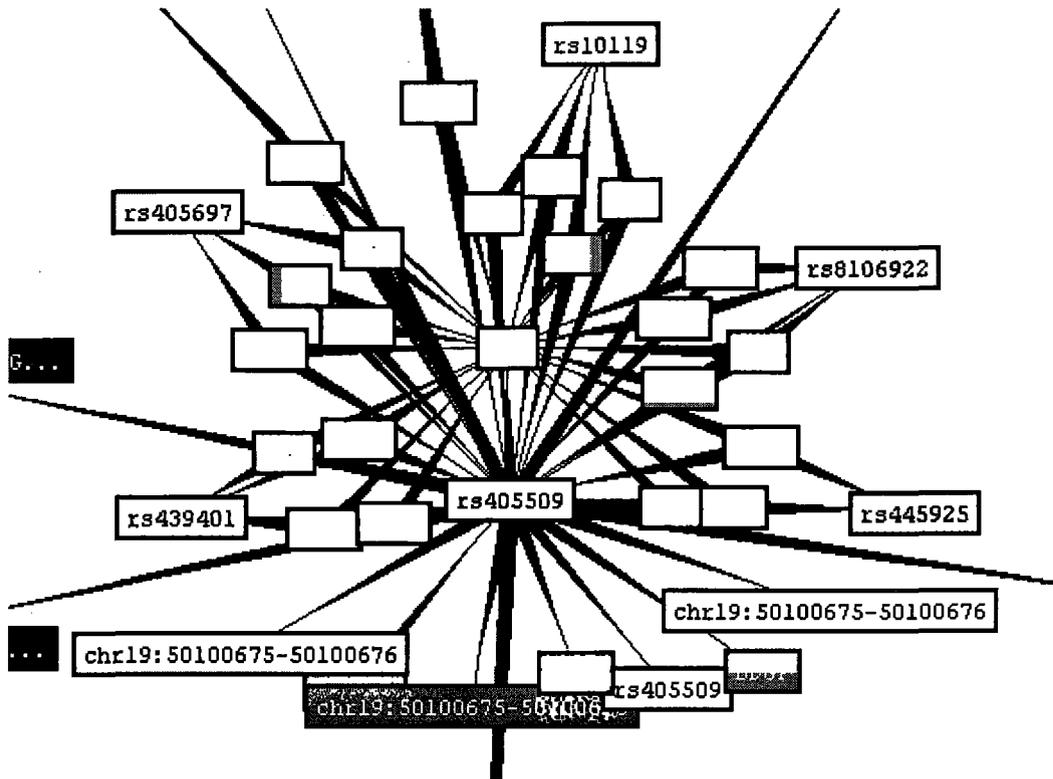


Figure 5.6. SNPit Touchgraph graphical user interface (152).

SNPit was also implemented as a text-based user interface which can be accessed through the Internet (Figure 5.7). Various iterations of the website were attempted in order to emphasize ease of use for potential users of the system. The main page of the website offers the ability to search either by SNP or by Gene. Within these two main groupings, there are a variety of entities that users can search by (Allele Frequency, Evolutionary Conservation, Gene Prediction, Linkage Disequilibrium, Population, Positive Selection, Protein Function Prediction, SNP, Tissue Expression, Gene, and Human Gene Mutation). The Main Search option

allows users to search all the data sources and entities at one time. The Rank SNPs option allows users to query a list of SNPs and view a list of SNPs ranked by SNP functional potential.

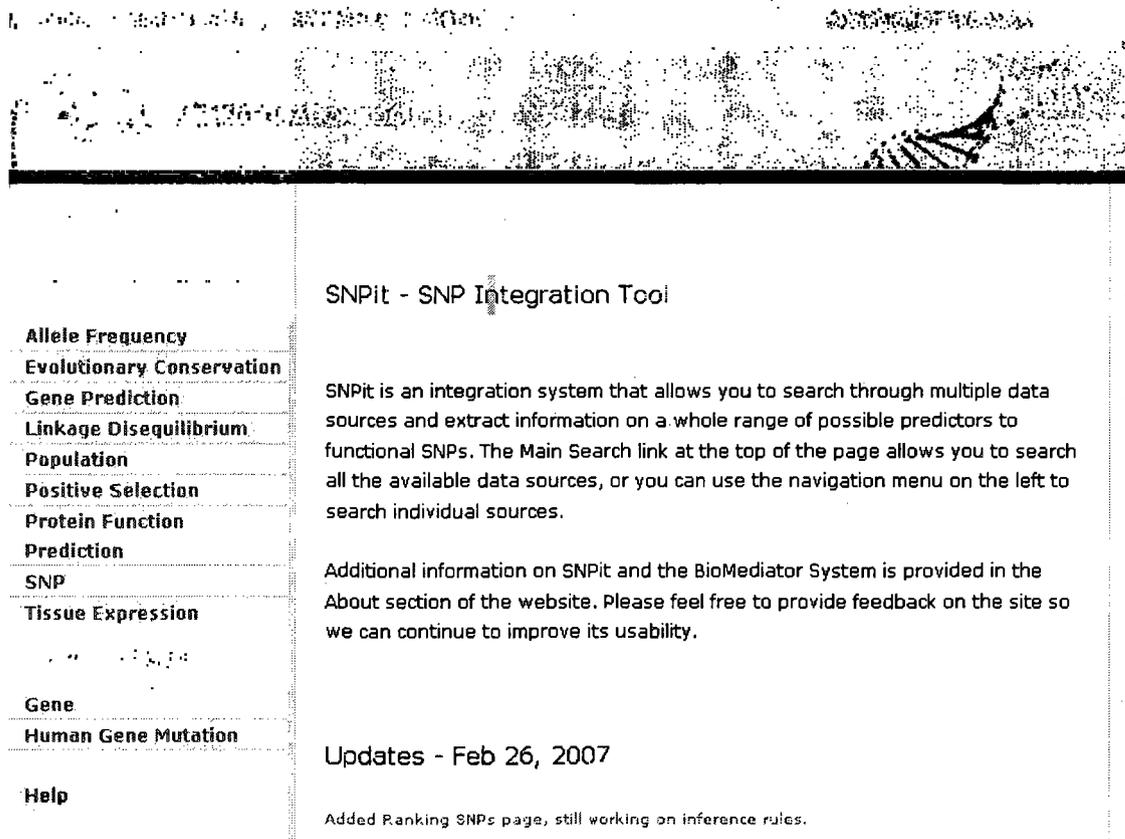


Figure 5.7. Overall design of the SNPit website (152).

5.4 Evolution of the SNPit System

The development of SNPit was iterative. In the first version of SNPit, we focused on the data integration component of the system, with no rankings involved.

In the second version of SNPit, we extended the federated data integration system to include logical inference for both point and regional SNPs. In the third version of SNPit, we extended the federated data integration system to include probabilistic inference for both point and regional SNPs. Finally, in the fourth version of SNPit, we extended the federated data integration to include both logical and probabilistic inference for both point and regional SNPs.

5.4.1 SNPit v1 – Data integration alone, no ranking

As mentioned previously in sections 5.1.1, 5.2.1, and 5.2.2, version 1 of the SNPit system encompassed the data sources, wrappers, common data model, and user interface extended onto the BioMediator system with the specific goal of SNP functional annotation. Each of the links in the SNPit system returns results in a text based format. For example, when the “Main Search” option is queried using rs405509 (Figure 5.8), the results for each of the different data sources are returned for that SNP (Figure 5.9). In this example, rs405509 returns information from dbSNP and HGMD. The dbSNP data source displays information on the chromosome location, gene id, gene name, observed alleles, predicted functional role, and validation method of the queried SNP. The HGMD data source provides information on the literature results for the gene that it is linked with, in this case, the Hypertension gene, APOE.

Home | About | Help | Search | Advanced Search | Contact Us

Search All Categories

This main searching page allows you to query all available data sources at one time. SNPs can be entered with or without the RS prefix (i.e. rs405509 or 405509)

Search: All Sources

Attribute: rs number

SNP IDs:

Search by Category

- Probabilistic Inference
- Probabilistic and Logical Inference
- Allele Frequency
- Evolutionary Conservation
- Gene Prediction
- Linkage Disequilibrium
- Population
- Positive Selection
- Protein Function Prediction
- SNP
- Splice Site Prediction
- Tissue Expression
- Transcription Factor

Figure 5.8. Input screen for main search option.

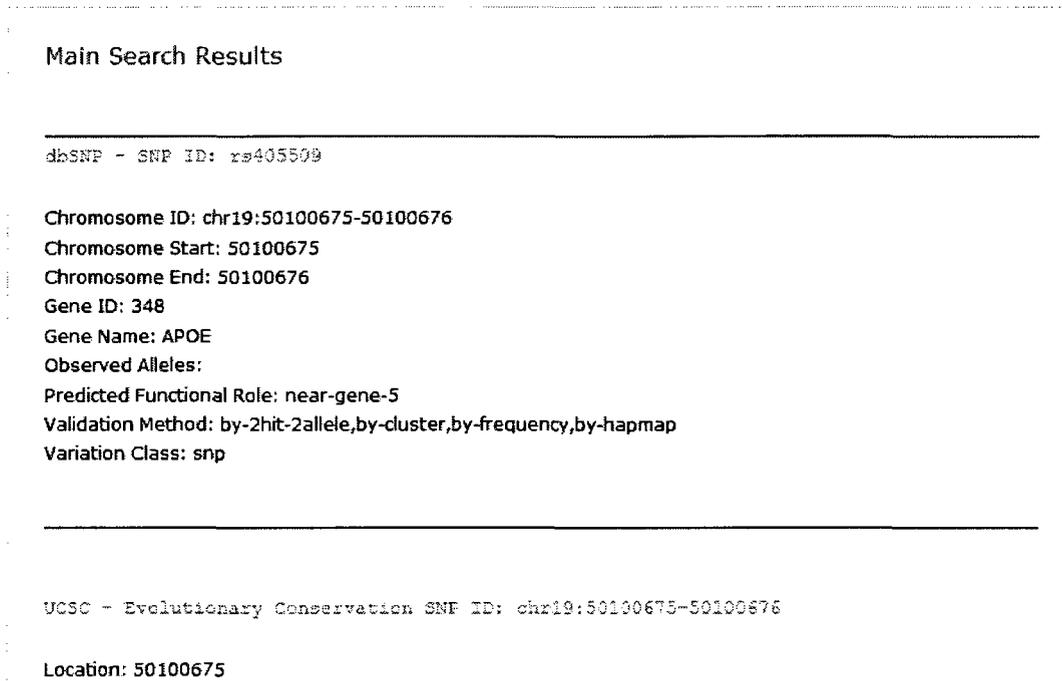


Figure 5.9. Returned results for SNP rs405509 using the Main Search option.

5.4.2 SNPit v2 – Ranked SNPs using logical inference

As mentioned in 5.2.4, the rules which were used to capture the expert knowledge in the decision tree (Figure X) was implemented using Jess rules. This can be accomplished in Jess through rules formation and facts creation. The objects in the mediated schema are first represented in Jess as an object, for example, to assert that a SNP is coding, we would use the expression:

```
(assert (coding_SNP mySNP))
```

Similarly, the rules were defined by creating `defrule` expressions. For example, in Figure 5.10 the name of the rule is `cSNP_nonsyn`, the antecedent of the rule is if the SNP is predicted to be coding-nonsynonymous, the consequent is to categorize this SNP as nonsynonymous and assign it a score. The complete Jess rules are in Appendix D – Jess rules implementing decision tree.

```

IF the SNP has a predicted functional role of coding-nonsynonymous,
THEN categorize this SNP as nonsynonymous and assign it a score.
(defrule check_cSNP_nonsyn
  (SNP (SourceID Temp_id) (PredictedFunctionalRole Ya:(regex Ya "coding-nonsynonymous")))
  (not(ProteinFunctionPrediction (SourceID Temp_id) (PredictionIdRole R)))
  =>
  (printout t "cSNP nonsyn" crlf)
  (assert (RankingSNP (remember Temp_id) (score (format nil "%.3f" 1.25)) (category "coding SNP,
nonsynonymous"))))
)

```

Figure 5.10. Jess rule for nonsynonymous SNPs (152).

A series of rules such as the above example are placed into working memory, when a SNP is queried, the rules are activated and new facts based on these rules are added to working memory as each rule is executed. The logical inference system then displays the results of these rules and scores. Figure 5.11 is the input screen of the SNPs that are entered into the logical inference rules ranking page. Figure 5.12 is a screenshot of the ranked SNPs based on logical inference.

Rank SNPs Page

This Rules Searching page allows you to apply rules to the SNPs you enter. SNPs can be entered with or without the RS prefix (i.e. rs405509 or 405509)

Search: All Sources

Attribute: rs number

SNP IDs: rs1098
rs1036939
rs2545
rs1113132
rs11037909
rs405509

Figure 5.11. Input screen for logical rules ranking page.

Rules Search Results		
SNP rs number	Rank Score	Ranking Determination
rs1098	3.375	coding SNP, nonsynonymous, damaging
rs1036939	2.250	non-coding SNP, 5' UTR
rs2545	1.500	non-coding SNP, 3' UTR
rs1113132	1.200	intronic SNP
rs11037909	1.200	intronic SNP
rs405509	1.000	non-coding SNP, non-UTR, flanking SNP, near-gene-5
rs1212171	1.000	non-coding SNP, non-UTR, flanking SNP, near-gene-5

Figure 5.12. SNPit display of ranked SNPs based on the logical rules implemented through Jess (152).

The Jess rules created for logical inference run individually for each SNP queried. To gain information on the region around a SNP, the SNP is first queried using the linkage disequilibrium option in the SNPit website and then the entire list can be run through the logical inference rules.

5.4.3 SNPit v3 – Ranked SNPs using probabilistic inference

As previously mentioned in section 5.2.3, probabilistic values were incorporated into SNPit . These Pr, Qr, Ps, Qs values were assigned based on the amount of trust we had in the individual records and source as well as the linkages present in records and sources. For version 3 of SNPit, we used these probabilistic values and adjusted the existing UII score algorithm to accomplish our goal of SNP prediction. Our SNP annotation network was different than the previous gene annotation network as described in section 4.4.2 in the last chapter. Rather than ranking the nodes based on the farthest result and its relationship to the initial seed query, in our case, we needed to rank the results based on the SNP entity and its attributes and relationships to surrounding nodes. Presented as an analogy, our SNP network comprised of various clusters or flowers which represented individual SNPs, each flower was then connected to other nodes which provided annotation clues to

that SNP. Thus, each flower had accompanying petals which described the inner SNP (Figure 5.13).

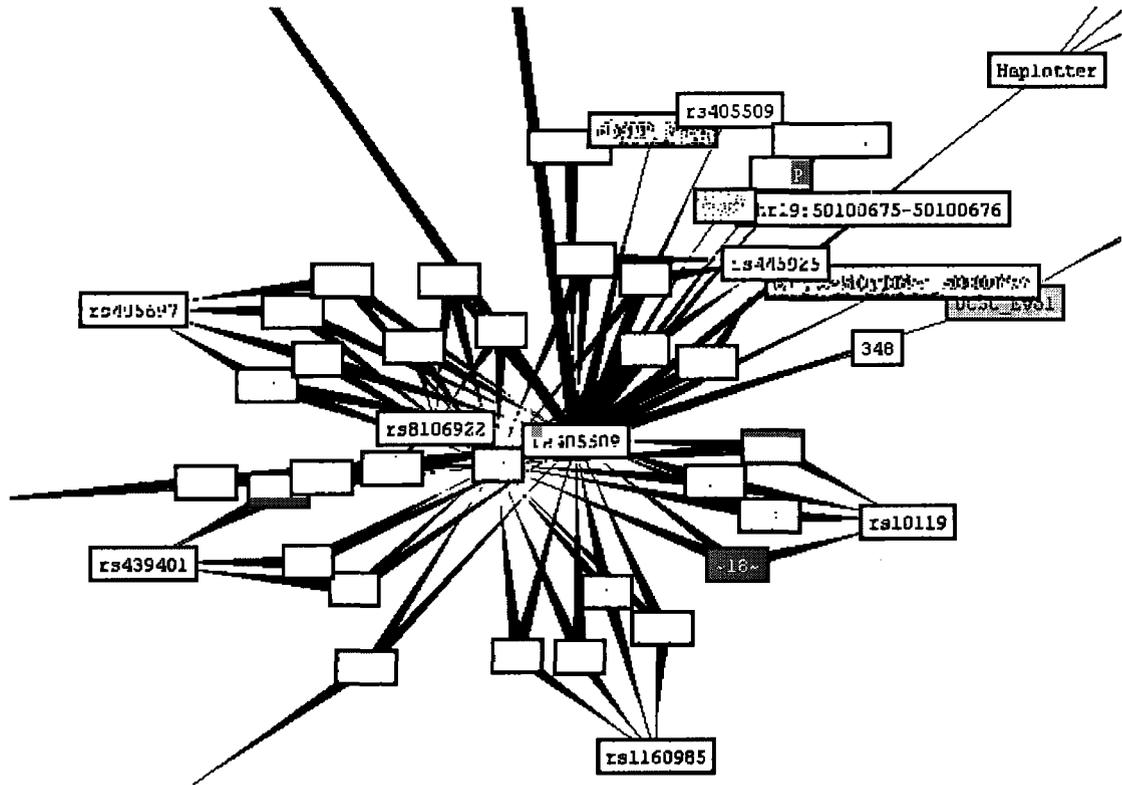


Figure 5.13. Central SNP presented as a flower analogy, with additional attributes acting as petals extending out from the central seed.

The integration of the probability scores was done using the following algorithm: first we averaged the related sources, then we calculated the maximum for the unrelated sources, and finally we averaged the independent sources. This process was done in a sequential manner, and produced a “SNP score” that was exclusive to just the SNP and its surrounding resources. This “SNP score” was then combined

with the original Pr score to get an overall Pr score. From there, the UII algorithm, based on the Monte Carlo algorithm was run, producing a final UII score. The following figure demonstrates the probabilistic scores for an example point SNP (Figure 5.14).

UII Probabilistic Results for Point SNP				
SNP rs number	Ps Score	Pr Score	SNP Probabilistic Score	SNP UII Score
rs405509	0.900	0.35	0.35	0.3116

Figure 5.14. Probabilistic results for SNPit.

The process was then repeated for regional SNPs, this required a slight modification to the mediated schema for logistical purposes. The regional probabilistic inference calculation calculates a “SNP score” for the queried SNP and those SNPs that are in linkage disequilibrium with that SNP. A cumulative Pr score is then produced and the UII algorithm is again applied. The following figure demonstrates the probabilistic scores for an example regional SNP (Figure 5.15):

SNPit UII Probabilistic for Regional SNP Page

SNP rs number	Ps Score	Pr Score	SNP Probabilistic Score	SNP UII Score
rs405697	0.900	0.375	0.375	0.0996
rs445925	0.900	0.375	0.375	0.047
rs5114	0.900	0.375	0.375	8.0E-4
rs439401	0.900	0.375	0.375	0.0932
rs10119	0.900	0.375	0.375	0.0856
rs8106922	0.900	0.375	0.375	0.1018
rs1160985	0.900	0.375	0.375	0.109
rs417357	0.900	0.375	0.375	0.0769
rs405509	0.900	0.35	0.35	0.3187

Figure 5.15. SNPit interface demonstrating the ranked SNPs that are returned when users select the Probabilistic Inference for Regional SNPs option.

5.4.4 SNPit v4 – Ranked SNPs using both logical and probabilistic inference

In version 4 of SNPit, we combined both the logical and probabilistic inference concepts in the previous two sections. This was accomplished by taking the previous Pr score which was modified with the customized algorithm that takes all the petals of the flower and pushes the values inward to come up with a “SNP score”, and modify that with the logical decision tree rules described earlier. This is done by first transforming the heuristic weights described previously in 5.2.4 from a range of 1 to 4 to a number between 0 and 1. This Jess calculation was then

imported into the SNPit UII plugin and multiplied with the original Pr score. Figure 5.16 demonstrates the ranked SNPs that are returned from the logical and probabilistic inference ranking page.

SNPit U2 Rules Page

SNP rs number	P's Score	Pr Score modified with heuristic weights	SNP Probabilistic Score	SNP UII Score
rs10119	0.900	0.445	0.445	0.058
rs8106922	0.900	0.356	0.356	0.0702
rs445925	0.900	0.356	0.356	0.0351
rs1160985	0.900	0.356	0.356	0.0873
rs417357	0.900	0.356	0.356	0.0339
rs405697	0.900	0.356	0.356	0.0744
rs439401	0.900	0.356	0.356	0.0747
rs5114	0.900	0.356	0.356	6.0E-4
rs405509	0.900	0.296	0.296	0.2656

Figure 5.16. SNPit interface demonstrating the ranked SNPs that are returned when users select the Probabilistic and Logical Inference for Regional SNPs option.

5.5 Discussion

Many lessons were learned in the four versions of SNPit described in this chapter. The baseline version of SNPit (SNPit v1) comprised of a federated data integration system with mediated schema, the application of a federated data integration system using BioMediator for the purposes of SNP annotation was found to be successful. The key finding was the validation of the BioMediator model in a third domain (functional SNP annotation). This was important since there were characteristics of SNP annotation that were quite different than protein and gene annotation which were the prior uses of the BioMediator system. The mediated schema required revisions as the system progressed. We also successfully implemented a web based text user interface that plugged into the BioMediator system. However, SNPit v1 does not return a list of SNPs ranked by potential function, it just returns the data from the various data sources.

In order to build inference into the system to prune and rank the result graph, SNPit v2 included logical inference in the form of rules. These logical rules based on SNP annotator expert knowledge were successfully implemented and Jess was used to instantiate these rules inside the Biomediator system. The key finding was the validation that the heuristics used for SNP annotation could be represented in a tractable computable form. The optimum design for the decision tree in the long term would require iterative design and modifications in the future.

Probabilistic inference was then added in SNPit v3, and we found that the UII algorithm could be applied to the SNP domain, but a key finding was that it

required a customized algorithm due to the result graphs of the SNPs being of a different topology than result graphs for protein annotation. The UII metrics for our SNPit system could also vary depending on the opinions of the SNP experts though this was of less concern due to the relative insensitivity to precise probabilistic scores of the probabilistic inference mechanisms.

Finally, we combined both logical and probabilistic inference for SNPit v4, and found that combining the UII algorithm with the Jess plug-in was achievable. An important finding was that due to the relatively orthogonal data needed by the two inference approaches there were multiple valid potential ways to combine the two forms of inference to generate a single pruned ranked list of results. The approaches we implemented and evaluated are described further in Chapter 6, section 6.4.4.

In this chapter, we discussed in detail the implementation of the base SNPit system and its subsequent evolution. We reviewed the four versions of SNPit: the federated data integration baseline system, the inclusion of logical inference, probabilistic inference, and finally, the combination of both logical and probabilistic inference. This chapter addresses the feasibility aspect of the overarching dissertation question. In the next chapter, we will examine how we evaluated the SNPit system when we were challenged with the lack of a gold standard.

Chapter 6: Assessing the Value of the SNPit System

6.1 Introduction

In the last chapter, we described the implementation of the SNPit system, including the different versions of the SNPit and what we learned about feasibility and implementation. In order to assess the value of the SNPit system, we were faced with the difficulty of coming up with an evaluation strategy due to two reasons: 1) there is no gold standard available for verifying the functionality of a SNP and 2) previous systems have not attempted to evaluate their results in a formal manner. These issues were first introduced in Chapter 3 section 3.5. In this chapter we describe both our evaluation approach (Chapter 6, section 6.4) and the results of our evaluation (Chapter 6, sections 6.4.1, 6.4.2, 6.4.3, 6.4.4, and 6.4.5).

6.2 Approach to gold standard and metrics

No true gold standard exists for the annotation of SNPs at the present. This is particularly the case for complex diseases where genetic factors only account for a portion the final phenotype in an individual. In order to come up with a reference standard if not gold standard for our evaluation, we used the database HGMD which provides published evidence on both SNPs that are statistically significant through

genome wide association studies as well as those that have functional evidence through in vivo techniques (HGMD details was discussed in section 5.3.1 of the previous chapter). HGMD was designated as an alternative gold standard.

6.2.1 Challenge to finding a gold standard

As stated in section 3.2 in Chapter 3, current systems for SNP annotation either do not perform an evaluation or select a set of SNPs from articles and run those SNPs as examples to describe the kinds of results their systems return. Finding a gold standard analogous to that used for typical laboratory tests is a challenge because the link from statistical significance to clinical significance has yet been made (see 2.3.1.2 on the whether or not genome wide association study results will ever attain clinical utility). When trying to find a gold standard for what the “true” association is, we looked at three possible databases. dbSNP is the largest repository of SNPs currently available. However, we determined that dbSNP included mainly false positives and most of the SNPs remain un-annotated and the curation of the SNPs is variable (47). Another possible database that was considered is the new NCBI database dbGAP, which is a new recent database that allows researchers to submit studies linking genotype to phenotype; however, since the database only started in 2008, there are currently only around 44 studies listed (174). We decided to use HGMD as our gold standard because they included both SNPs that have been

statistically linked to a clinical phenotype as well as those that have expression/transcriptional/splicing evidence. Even more importantly, HGMD only includes information in their database after manual curation has demonstrated some kind of functional evidence (164). Thus, HGMD acted as a source of true positives for evaluation purposes.

Similarly, we settled on dbSNP as a source of true negatives. For dbSNP's most recent version (build 129), there was a total of approximately 20M SNPs with rs numbers for *Homo sapiens*, and only 17 thousand of these had some kind of functional class assigned to it and were cited in PubMed. This indicates that the majority of SNPs found in dbSNP would not be functionally relevant. Therefore, we decided to use dbSNP as our source of true negatives.

In addition to the issue of finding a gold standard, we wanted to get an understanding of how common SNPs and thus, common diseases, are measured using SNPit. As a preliminary attempt, we gathered a collection of SNPs that have a minor allele frequency of greater than 5 percent and ran those through SNPit. After conducting an initial test on 200 SNPs within HGMD, it was concluded that there was not enough common SNPs to be able to narrow down this evaluation section to just common SNPs (as mentioned in Chapter 3, section 3.5, there are only 240,000–400,000 common cSNPs out of a total estimated 11 million SNPs in the human genome). However, we can still proceed with the evaluation, albeit without

the limitation of only looking at common SNPs. That would result in two lists, the SNPs that score as positive in SNPit and those that score as negative in SNPit.

6.2.2 Challenge to evaluating clinical significance

As discussed in 2.3.1 in Chapter 2, currently there is a significant gap between a genome wide association study result that is statistically significant and one that is clinically significant. Furthermore, there is still much to be learned on how results from a genome wide association study might effects gene-gene and disease-disease interactions: a variation that changes the function of one gene might impact other genes that in the vicinity; similarly, SNPs that have an effect on one disease might impact other diseases as well. Statistical results from genome wide association studies have been modest and understanding the biological mechanisms behind the results have been a continuing challenge. Numerous functional loci have yet to be discovered and the impact of the environment in conjunction with genetic variation has been an area of research that is only beginning. Because there is no clear metric to measure if a result is clinically significant and because of all the additional areas of research into genotype-phenotype correlation that is needed, there is no “gold” standard catalogue of SNPs that are definitively causal.

We will not be conducting an evaluation of clinical significance in this dissertation. Our evaluation is restricted to how accurate our SNPit system is at predicting functional SNP annotations. The road to confirming clinical significance

would be restricted to a particular study/disease and would entail first conducting a clinical validity test as described in 2.3.1.2 of Chapter 2, as well as a general background understanding on how important the trait being investigated might be and a calculation of the population attributable risk of the study.

6.2.3 What metrics were chosen and why

Once we selected an alternative gold standard in the form of HGMD, we needed to select metrics that would help us evaluate the accuracy of the SNPit system. For this portion of the evaluation, two approaches were taken. For the baseline version of SNPit where a federated data integration system was created for SNP annotation, but where no ranked SNPs were created, we decided to use sensitivity and specificity as the measures of analysis. Sensitivity and specificity are standard performance measures for binary classification when there is no ranked list. For the versions of SNPit that produce a ranked list of SNPs, we decided to use recall and precision, which are also standard classification measures, but also used in information retrieval when a ranked list of results are returned.

6.2.3.1 Sensitivity/Specificity

Sensitivity is defined as the ability to detect the positive evidence for SNP functionality, and specificity is defined as the ability to detect non-evidence for SNP

functionality. A true positive (TP) result would be when the test is correctly classified as functionally relevant, a true negative (TN) result would be when the SNP is correctly classified as being non-functional; false positive (FP) would be when the SNP is incorrectly classified as being functional, and false negative (FN) would be when the test incorrectly classifies the SNP as being non-functional (175).

The formula for sensitivity is: $100 \times TP / (TP + FN)$, we're using HGMD to provide the information for the true positives and false negatives. The formula for specificity is: $100 \times TN / (TN + FP)$, we're using dbSNP (in particular, those SNPs that are found in dbSNP but not found in HGMD), to provide the information for the false positives and true negatives. Therefore, we will be testing the SNPs by its category. Non-synonymous SNPs are tested against SNPit's SIFT category, regulatory SNPs are tested against SNPit's TFSearch category, and splice site SNPs are tested against SNPit's BDGP category.

There are pros and cons to the use of sensitivity and specificity in terms of suitability for SNP annotation. Both of these measures possess the ability to differentiate between different measures and can be used in sensitivity analyses. However, the quality of sensitivity and specificity measures depend on a variety of factors including levels of cutoff and the quality of the data sources themselves (175). Furthermore, sensitivity and specificity would not be suitable once we started ranking SNPs and testing the accuracy of one list of ranked SNPs against another.

In addition, we will also be calculating the sensitivity and specificity of the entire federated system. Rather than separating the SNPs being tested by category as done during the first phase of the evaluation, we will be testing how the entire federated system returns annotation information on tested SNPs. Thus, in the second phase of the evaluation, we will be examining whether or not any of the SNPs are flagged by any of the categories as functionally promising. Thus, we will test HGMD against all three categories: non-synonymous, regulatory, and splicing, with true positives defined as those that are flagged as interesting by at least one of the three categories. Likewise, we will run those SNPs that are in dbSNP but not in HGMD against all three categories to establish the false negatives.

6.2.3.2 Precision/Recall

In the second phase of our evaluation, we used precision and recall as the main evaluation metric to assess accuracy. Used primarily in the field of information retrieval, precision is defined as the number of correct records retrieved divided by the number of records retrieved; recall is defined as the number of correct records retrieved divided by the total number of correct records in the complete collection (175). Disadvantages to using precision and recall include the fact that our user base might not be interested in these measures. Furthermore, recall requires that we know the total number of correct the records in the complete collection, which might be a

Figure 6.1. Recall and precision measured using HGMD and dbSNP.

6.2.3.3 Accuracy

The measures of sensitivity/specificity/recall/precision are built on the same basic categories of true positive, false positive, false negative, and true negative (Table 6.1). Using these measures, we were able to calculate accuracy for both phases of our evaluation. Accuracy tells us the percentage of predictions that are correctly predicted (Table 6.2).

		Predicted Label	
		Positive	Negative
Known Label	Positive	True Positive (TP)	False Negative (FN)
	Negative	False Positive (FP)	True Negative (TN)

Table 6.1. Table of TP, FN, FP, TN for classifier systems performances (176).

Measure	Formula	Intuitive Meaning
Precision	$TP / (TP + FP)$	The percentage of positive predictions that are correct.
Recall/Sensitivity	$TP / (TP + FN)$	The percentage of positive labeled instances that were predicted as positive.
Specificity	$TN / (TN + FP)$	The percentage of negative labeled instances that were predicted as negative.
Accuracy	$(TP + TN) / (TP + TN + FP + FN)$	The percentage of predictions that are correct.

Table 6.2. Various formulas that can be created by using the labels in the table of True Negatives, False Positives, False Negatives, and True Positives (176).

6.3 Calibration Analysis of some of the data sources

For the within and across categories sensitivities and specificities, the cutoff for the SIFT category was chosen somewhat arbitrarily, and for the splice sites and transcription factor binding sites, it was a binary choice, whether or not it caused a difference in the scores. For all these categories, we needed to figure out what was the current cutoff score and difference in score should be. Also, we needed a way to normalize the scores across all three categories. Thus, we decided to conduct a calibration analysis to examine these issues.

6.3.1 Calibration analysis for nonsynonymous/synonymous, splicing, and transcription factor categories

For the SIFT category, we ran hundreds of SIFT HGMD records from candidate genes and then we ran hundreds of completely random coding SNPs from dbSNP into SNPit. We then were able to come up with the following calculations.

		405			876	
		0.652671			0.127218	
0.02	0.638037	756	0.02	0.15	935	1.747984
		0.671755			0.159763	
0.03	0.662577	725	0.03	0.15	314	1.432612
		0.694656			0.168639	
0.04	0.693252	489	0.04	0.15	053	1.40348
		0.700381			0.186390	
0.05	0.699387	679	0.05	0.175	533	1.280281
		0.721374			0.198224	
0.06	0.717791	046	0.06	0.175	852	1.239929
		0.736641			0.207100	
0.07	0.723926	221	0.07	0.175	592	1.211906
		0.744274			0.213017	
0.08	0.736196	809	0.08	0.175	751	1.190452
		0.751908			0.221893	
0.09	0.742331	397	0.09	0.225	491	1.154555
		0.774809			0.224852	
0.1	0.773006	16	0.1	0.225	071	1.174065
		0.853053			0.310650	
0.2	0.834356	435	0.2	0.325	888	0.935617
		0.881679			0.393491	
0.3	0.858896	389	0.3	0.425	124	0.763432
		0.891221			0.452662	
0.4	0.871166	374	0.4	0.5	722	0.670819
		0.906488			0.505917	
0.5	0.895706	55	0.5	0.6	16	0.610488
		0.933206			0.547337	
0.6	0.92638	107	0.6	0.625	278	0.580921
		0.942748			0.571005	
0.7	0.932515	092	0.7	0.625	917	0.562535
		0.952290			0.609467	
0.8	0.944785	076	0.8	0.65	456	0.532369
		0.959923			0.647928	
0.9	0.957055	664	0.9	0.65	994	0.504782
1	1	1	1	1	1	0.340717

Table 6.3. Spreadsheet of how we arrive at the calculation of the probability of B|A, which equates to the probability of a SNP being found in HGMD given a change in its score.

The probability of B given A is the probability that a random SNP from dbSNP is in HGMD given some score for that SNP in dbSNP. When the cutoff scores from SIFT

are graphed against the probability of HGMD, the results can be used to find the score that maximizes the probability of being found in HGMD.

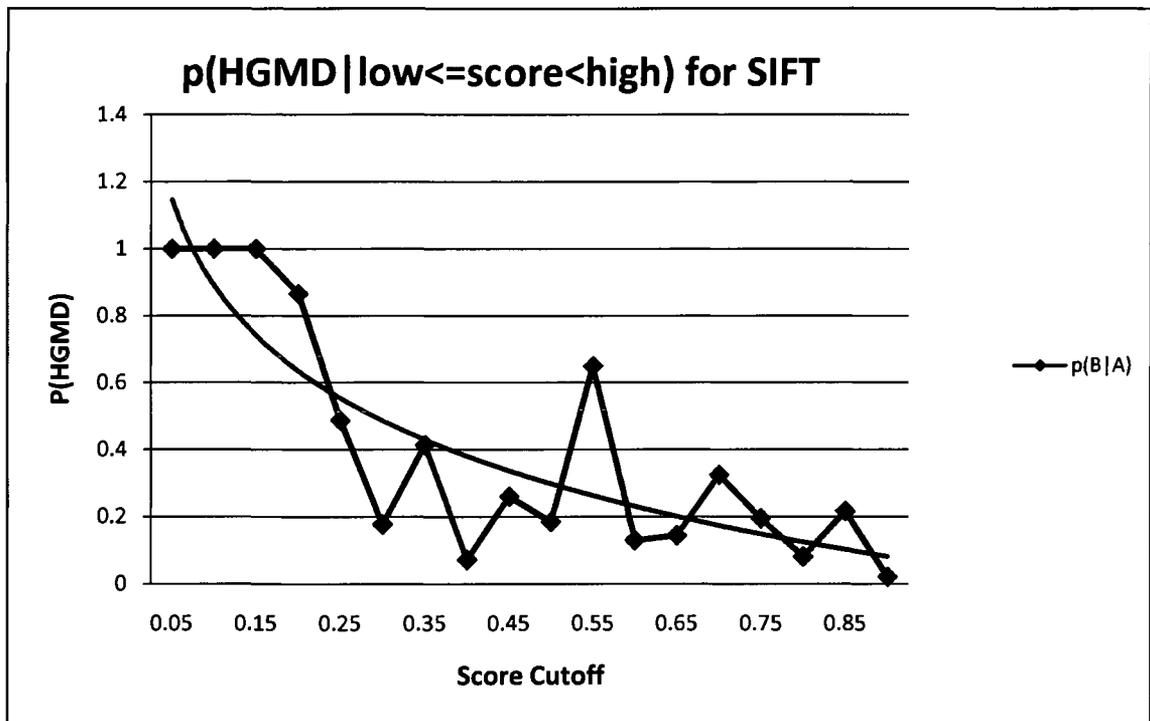


Figure 6.2. Graph of the probability of a random SNP from dbSNP being in HGMD given some score for the SIFT category.

For the SIFT category, the graph (Figure 6.2) demonstrates that a random SNP from dbSNP is in HGMD given some score is highest at the score cutoff of 0.55. We repeated this same process for the other two categories. For the BDGP category, the score cutoff difference with the highest $P(\text{HGMD})$ level is 0.55 (Figure 6.3). For the TFSearch category, the score cutoff difference with the highest $P(\text{HGMD})$ level is 0.25 (Figure 6.4). The probability of HGMD was used as the

main source of comparison between the three categories. We translated SNP's score cutoff or score difference from missense/nonsense, splicing, and transcription factor binding to its P(HGMD) score.

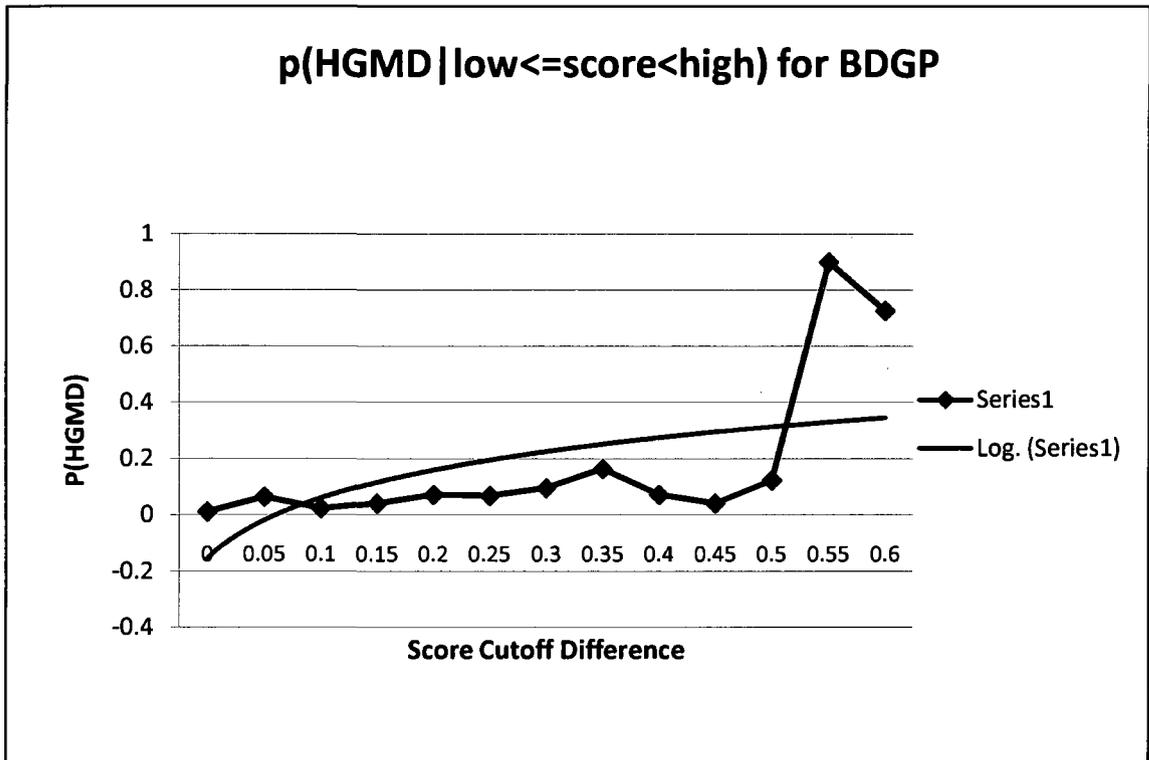


Figure 6.3. Graph of the probability of a random SNP from dbSNP being in HGMD given some score for the BDGP category.

These graphs were later revised for normalization from a 0 to 1 scale and a log function was also applied to the results, the log function allowed us to observe the general trendlines of the data. From there, we were able to take the results and revise the Pr scores for the Protein Function Prediction (SIFT), Transcription Factor

Binding (TFSearch), and Splice Site (BDGP) entities. The Pr scores were revised so that they would now reflect the P(HGMD) value for the log function in each graph. This would capture the predictive power of the records retrieved from these three sources more accurately than before.

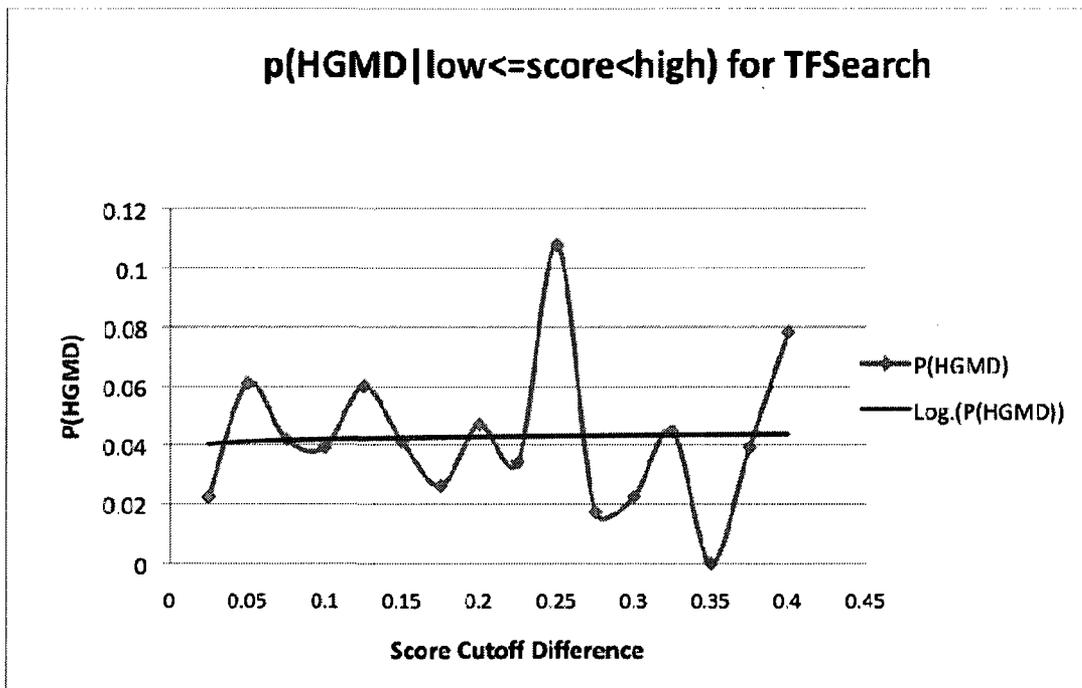


Figure 6.4. Graph of the probability of a random SNP from dbSNP being in HGMD given some score for the TFSearch category.

6.4 Evaluating the different versions of SNPit

The different versions of SNPit which was discussed previously in section 5.4 of Chapter 5 were evaluated using either sensitivity/specificity or

recall/precision, metrics which were discussed in section 6.2.3. Accuracy was calculated across all versions as one common metric.

6.4.1 Evaluating SNPit v1 – Data integration alone

To measure the sensitivity and specificity of the baseline SNPit system with just the data integration component, we used dbSNP as the source of true negatives and HGMD as the source of true positives. For the within category sensitivity and specificity, we looked at three categories (synonymous/nonsynonymous, splicing, transcription factor binding) separately for five different candidate genes (CFTR, BRCA1, BRCA2, NF1, TP53). For each category, hundreds of SNPs were run through the SNPit system.

6.4.1.1 Categories used in this evaluation

Types of functional SNPs was first presented in 3.2 of Chapter 3. The categories that were used during the first phase of evaluation were: Synonymous/Nonsynonymous, Splicing, and Transcription Factor Binding. These categories are how gene lesions are listed in the data source HGMD (for details on the data source, see Chapter 5, section 5.3.1). We evaluated five candidate genes: CFTR, BRCA1, tp53, NF1, BRCA2 using the evaluation techniques described below (sections 6.4.1.2 and 6.4.1.3).

6.4.1.2 Category by category evaluation

(Synonymous/Nonsynonymous, Splicing, and Transcription Factor Binding Separately), using sensitivity and specificity

For the missense/nonsense category of a particular gene, we retrieved the mutations from HGMD, performed some manipulations so that were properly formatted for SIFT analysis, and then found the corresponding conditional probabilities that predict whether the change in amino acid detrimentally affects the protein. For the probabilities that are low, that would suggest that a change in amino acid would be detrimental. We then chose a threshold (0.2) to determine which SNPs would be scored as a true positive. For the true positives category, we found the unique sequences of those SNPs that are in the same candidate gene as before, and found the corresponding sequences for those rs numbers and submitted them to SIFT. The same threshold was selected for determining which SNPs would be deleterious.

For the splicing category of a particular gene, again we started by retrieving splicing mutations from HGMD. Manipulation of the data was necessary to find the corresponding sequences. The sequences were then run through the BDGP section of SNPit. BDGP uses a neural network to predict the acceptor and donor predictions scores for the sequence entered. For two sequences that have one nucleotide

difference between them, we then analyzed the returned acceptor and donor site prediction scores. If a difference is detected in the acceptor and donor site scores due to the change in a SNP in sequence, then we deemed that a true positive. For the true negatives, we found all the SNPs in that gene which are in dbSNP, but not in HGMD, and repeated the above procedures. Table 6.4 presents a subset of the data retrieved for this phase of the evaluation.

For the transcription factor binding category of a particular gene, we retrieved the regulatory mutations from HGMD. The sequences were run through the TFSearch component of SNPit. TFSearch uses a correlation calculation to predict transcription factor binding sites for the sequences entered. If the submitted sequences produce a change in the transcription factor binding site predictions, then we scored them as being a true positive. Again, we repeated the process for those SNPs that are in the same gene that can be found in dbSNP, but not in HGMD.

When the sensitivities and specificities of each of these categories were graphed (Figure 6.5), we were able to see that the synonymous and splicing categories have a higher level of sensitivity and specificity as compared to the regulatory category, demonstrating the weaker predictive ability of that particular data source.

	TP	FN	TN	FP
NF1	182	60	1492	254
tp53	24	6	520	110
BRCA1	200	52	968	122
BRCA2	62	16	1374	258
CFTR	190	74	1595	166
TOTALS	658	208	5949	910

Table 6.4. Splicing category evaluation for five candidate genes, pertains to version 1 of SNPit (177).

Category by category evaluation

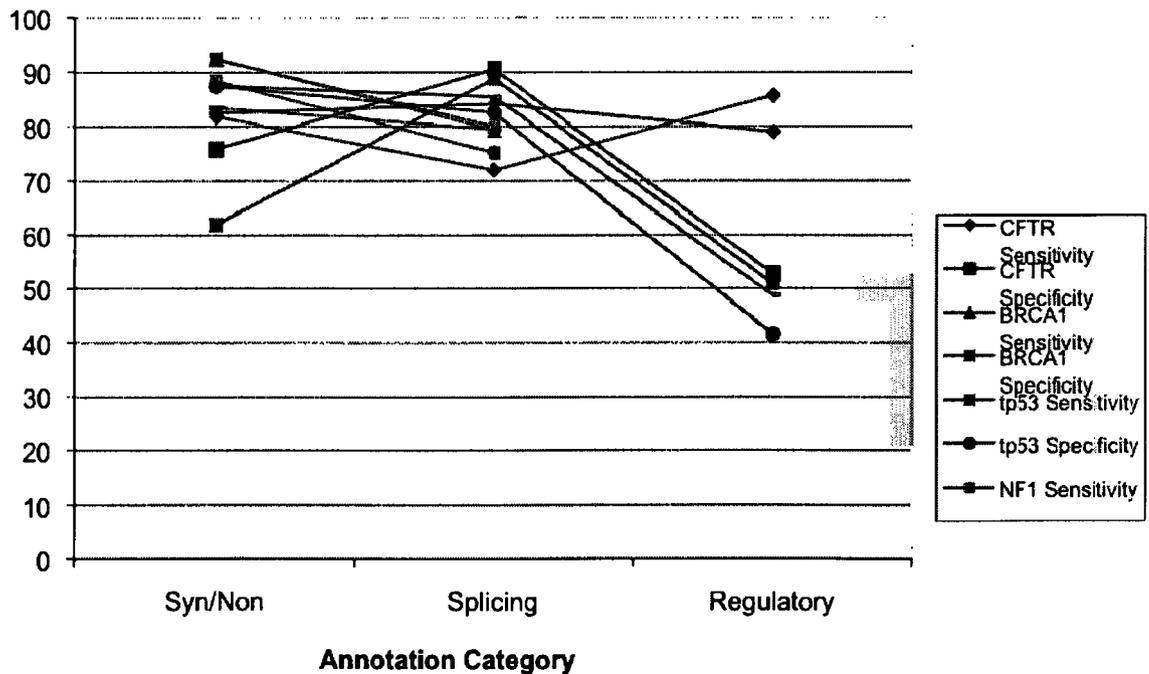


Figure 6.5. Figure demonstrates the final results of all the sensitivities and specificities of the candidate genes, pertains to evaluation of SNPit v1 (177).

6.4.1.3 Aggregate cross category evaluation (Combining Synonymous/Nonsynonymous, Splicing, and Transcription Factor Binding), using sensitivity and specificity

The general idea for this second cycle of evaluation is to compare each category with every other category. The table below demonstrates this idea (Table 6.5). For the missense/nonsense category for example, the cell entitled “SIFT_Missense/Nonsense” would be the evaluation that occurred during the previous within category sensitivity and specificity. We took the HGMD mutations for missense/nonsense and submitting it to SIFT missense/nonsense. However, for this second cycle, we also looked at the HGMD missense/nonsense mutations and how they impacted the splice site category. We also looked at the HGMD missense/nonsense mutations and their impact on transcription factor binding sites. For each row, if there was at least one positive hit for any of the three categories, it was counted as a true positive; otherwise the SNP was counted as a true negative. We then repeated this process for the other two categories.

HGMD Mutations	SIFT	SNPit Splice Site	SNPit TF Search
Missense/Nonsense	SIFT_Missense / Nonsense	BDGP_Missense / Nonsense	TFSearch_Missense / Nonsense

Splicing	SIFT_Splicing	BDGP_Splicing	TFSearch_Splicing
Regulatory	SIFT_Regulatory	BDGP_Regulatory	TFSearch_Regulatory

Table 6.5. The following table demonstrates how the sensitivities and specificities for the across category evaluations compare to one another.

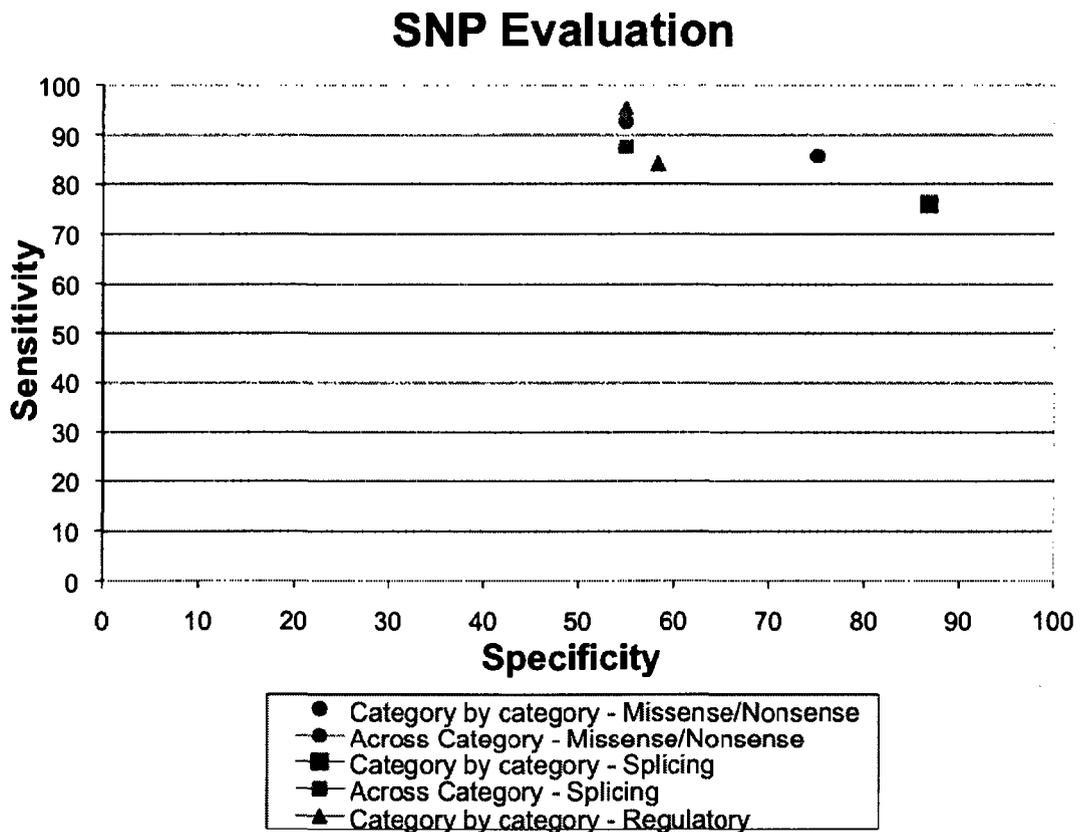


Figure 6.6. Both cycles of the first phase of evaluation for the missense/nonsense, splicing and regulatory categories (177).

The graph of sensitivity versus specificity for both cycles demonstrated that using data integration increases the sensitivity (Figure 6.6). This increase in sensitivity is due to the fact that we combined all three categories for cycle 2. For example, those SNPs that were categorized as regulatory might have been miscategorized, and by combining all three categories, we are able to catch more of the false negatives.

6.4.2 Evaluating SNPit v2 – Ranked SNPs using logical inference, using recall and precision measures

SNPit v2 produced a list of ranked SNPs using logical inference (Chapter 5, 5.4.2). In order to evaluate the recall and precision of this ranked list of SNPs, we ran 250 random SNPs from HGMD and 250 random SNPs from dbSNP. The decision tree scores from SNPit for both of these two sources of true positives and true negatives were recorded. From there, recall and precision measures were taken at 50 level intervals. In order to assess the predictive or discriminatory power of SNPit v2 using logical inference, a receiver operating characteristic (ROC) analysis was also performed. The ROC curve was created using the true positive (or sensitivity) and false positive rates (or $1 - \text{specificity}$) plotted along a graph; the area under the curve (AUC) measures the overall predictive power of the instrument

being measured, an ROC curve approaching the upper left hand corner would indicate a good outcome (178, 179).

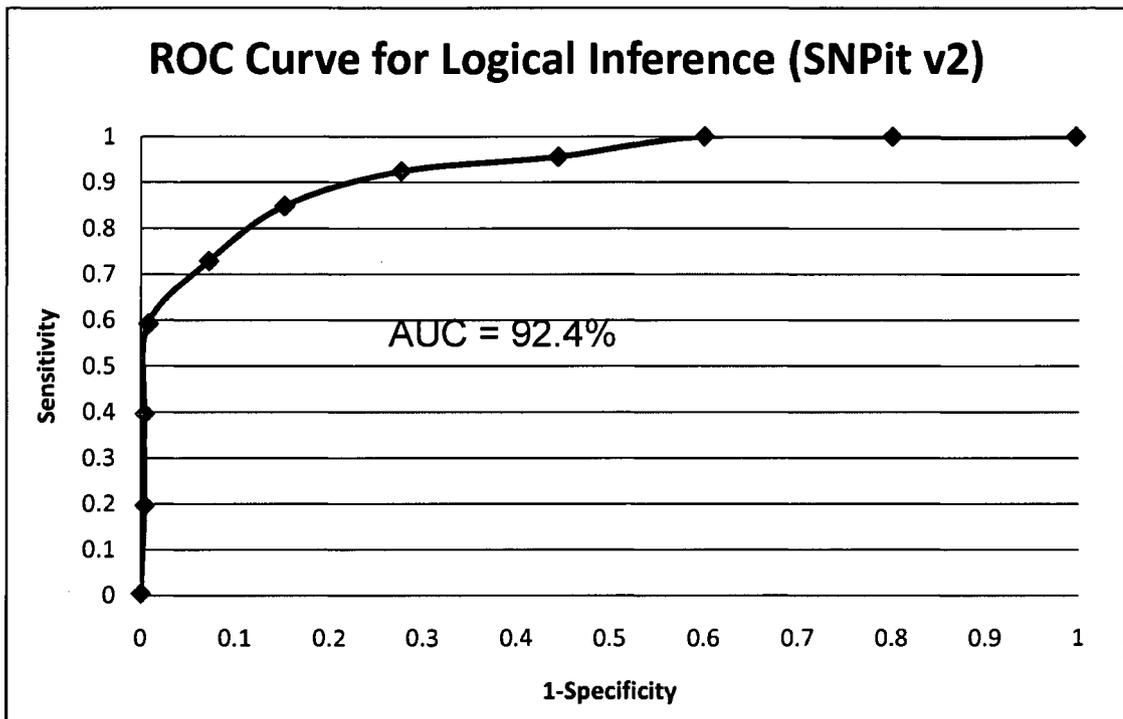


Figure 6.7. ROC curve for SNPit v2 – logical inference, indicating good predictive power because the graph line curves along the upper left corner, the area under the curve was 92.4%.

For the evaluation of SNPit v2, the ROC curve indicated a very good performance in terms of predictive power (Figure 6.7). The trapezoid rule was then used to calculate the area under the curve, with SNPit v2 having an area under the curve of 92.40%. We conducted a similar analysis to arrive at a precision recall graph, which is very similar to a ROC curve, only plotting precision versus recall. The results are

counterpart to the ROC curve, except that the precision recall graph is used more frequently in information retrieval (Figure 6.8).

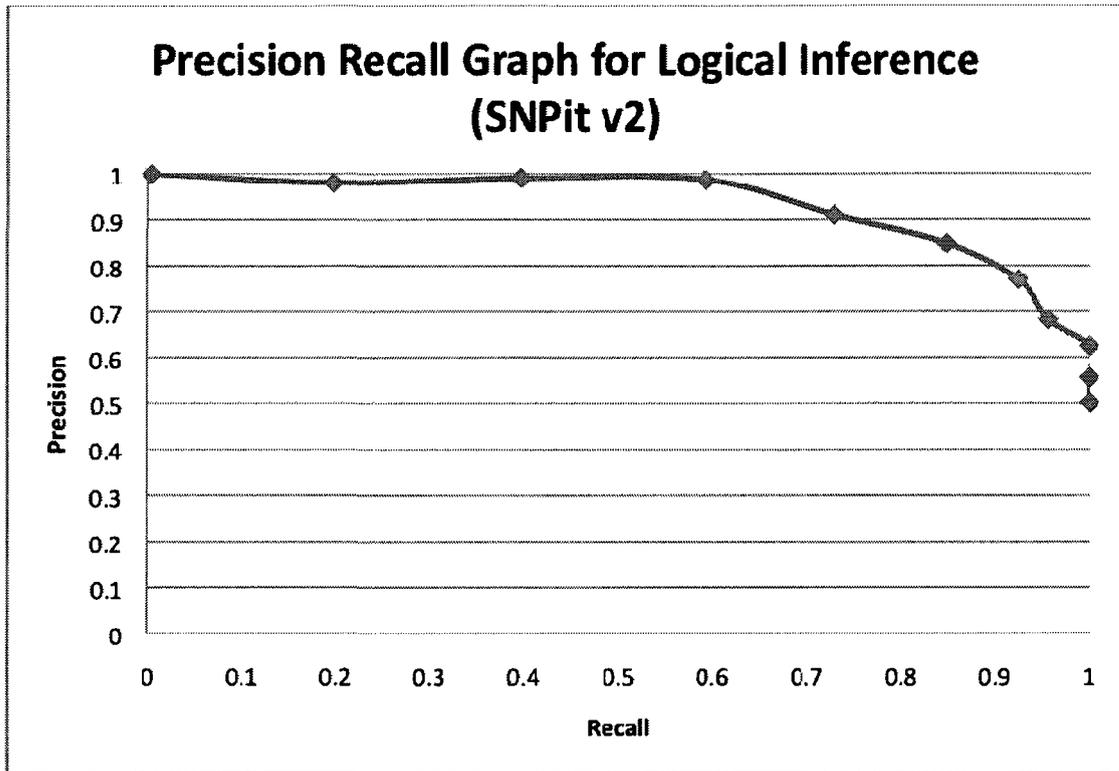


Figure 6.8. Recall/precision curve for SNPit v2 – logical inference, indicating good predictive power because the graph line curves along the upper right corner.

Examining the classification results of the SNP functional categories that make up the logical inference rules (Figure 6.9) reveals that the majority of SNPs that were randomly selected from HGMD come from “coding SNP, nonsynonymous” as well as “coding SNP, nonsynonymous, damaging” SNPs. Whereas SNPs that are randomly selected from dbSNP are mainly categorized as “intronic, low evolutionary conservation” and “intronic, high evolutionary

conservation”. This helps to explain why the logical inference version of SNPit performs well in the ROC curve for SNPit v2, more of the SNPs that are randomly chosen in HGMD are nonsynonymous than those randomly chosen in dbSNP. This is confirmation of general biological knowledge that exonic SNPs, particularly those that are nonsynonymous, are more likely to be functionally significant than SNPs in intronic regions.

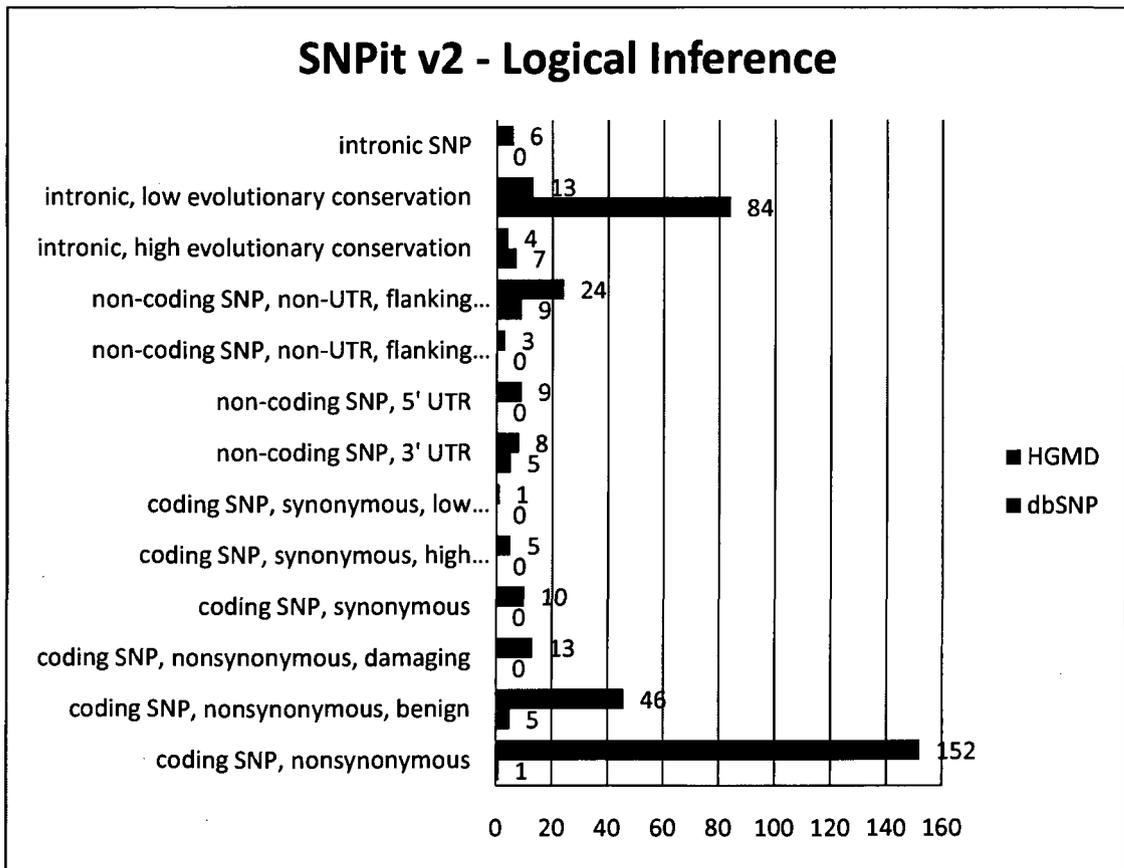


Figure 6.9. Classification of the SNP functional groups for logical inference rules separated by SNPs randomly run through HGMD and dbSNP

6.4.3 Evaluating SNPit v3 – Ranked SNPs using probabilistic inference using recall and precision measures

SNPit v3 produced a list of ranked SNPs using probabilistic inference (Chapter 5, 5.4.3). In order to evaluate the recall and precision of this ranked list of SNPs, we ran 250 random SNPs from HGMD and 250 random SNPs from dbSNP. The decision tree scores from SNPit for both of these two sources of true positives and true negatives were recorded. From there, recall and precision measures were taken at 50 level intervals. In order to assess the predictive or discriminatory power of SNPit v3 using probabilistic inference, a receiver operating characteristic (ROC) analysis was also performed. The ROC curve for the probabilistic inference version of SNPit performed moderately well, with the graph line extending further along the upper left hand corner than the diagonal, which would have indicated random predictability (Figure 6.10). The AUC for SNPit v3 was 68.11%.

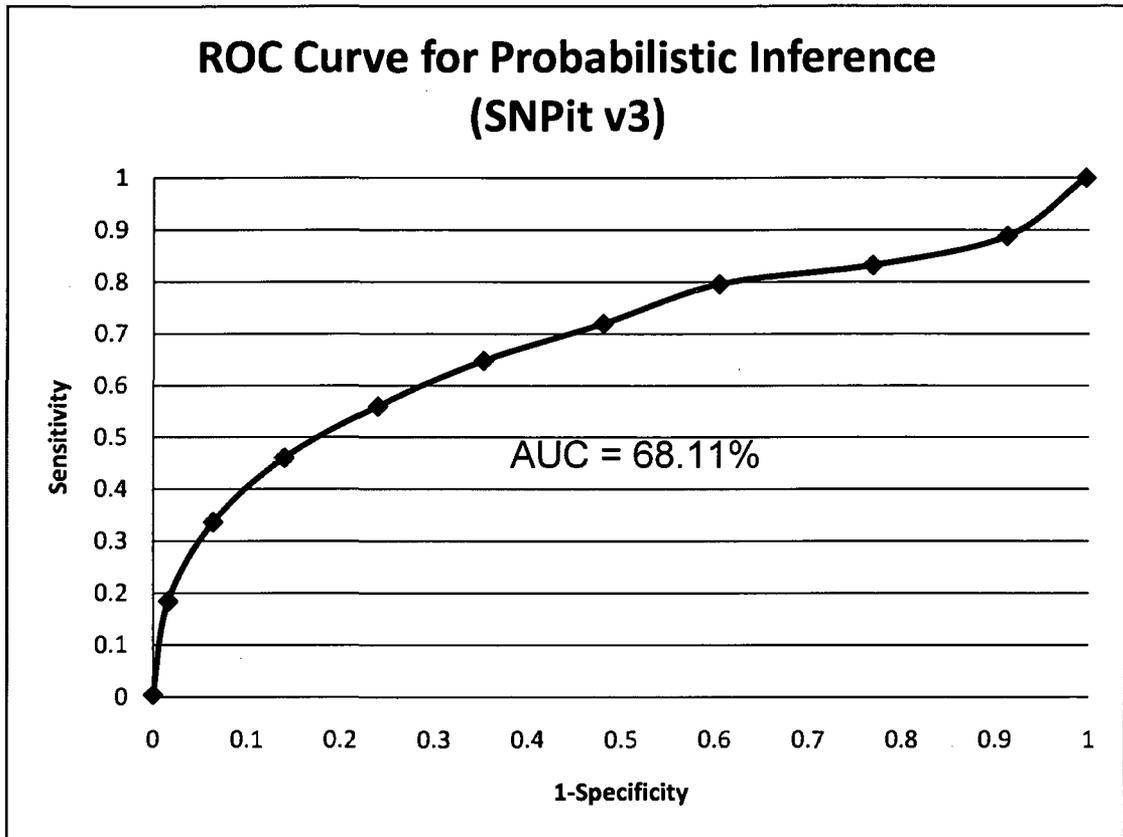


Figure 6.10. ROC curve for SNPit v3 – probabilistic inference, indicating moderately good performance because the graph line extends beyond the diagonal towards the upper left corner of the graph, the area under the curve was 68.11%.

Examining the break-down of the types of functional information that contributed to the “SNP Score” (described in Chapter 5, 5.4.3) for the probabilistic inference component of this evaluation, the characteristics of these data sources reveal that SNPs that randomly selected from HGMD have a higher proportion of SIFT scores and a slightly higher level of BDGP scores than random SNPs selected from dbSNP (Figure 6.11). The number of SNPs randomly selected from dbSNP and HGMD were approximately equivalent in terms of providing information on evolutionary

conservation, transcription factor binding, and linkage disequilibrium. This would explain the moderate improvement in predictability since only 2 out of the 5 categories increased the SNP score which is then converted into the probabilistic inference score (UII score).

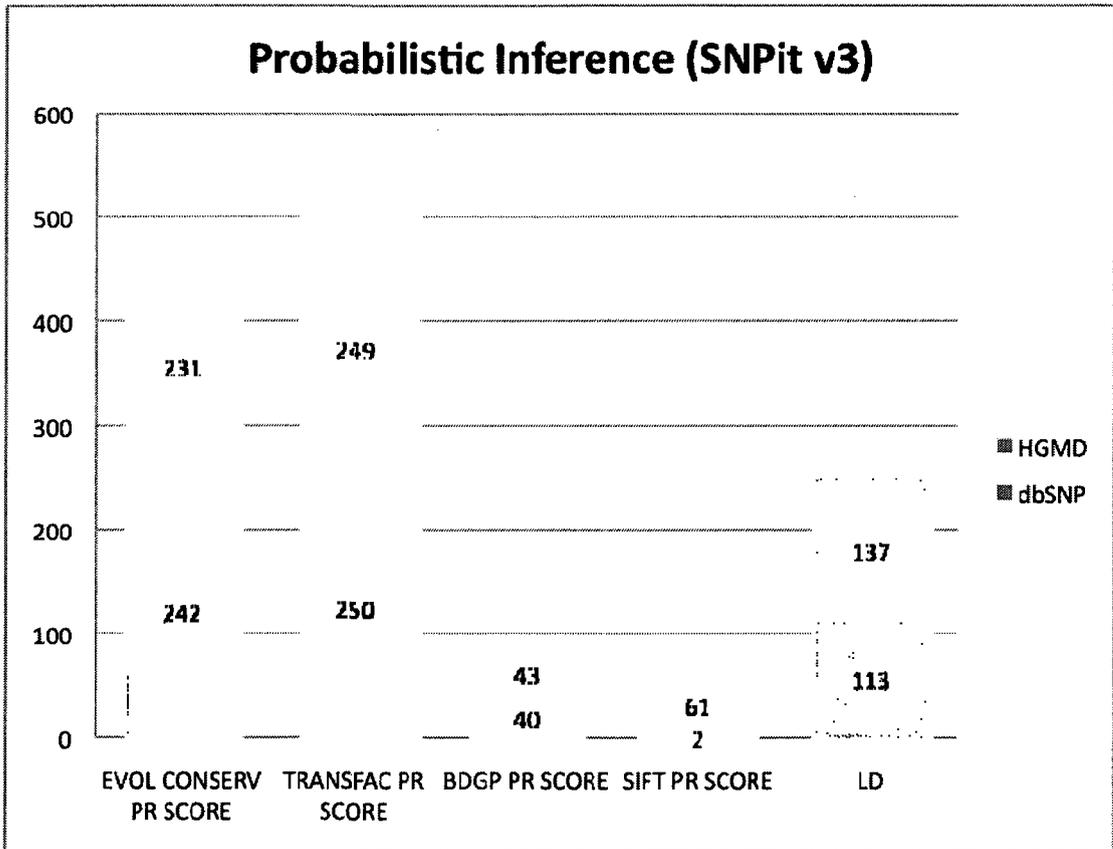


Figure 6.11. Breakdown of the different categories that provided information to the SNP score, which contributed to the probabilistic inference algorithm.

6.4.4 Evaluating SNPit v4 – Ranked SNPs using both probabilistic and logical using recall and precision measures

For the final version of SNPit, we decided to combine the probabilistic and logical inference approaches (detailed in Chapter 5, 5.4.4). Evaluation of the probabilistic and logical inference version of SNPit followed the same procedure as SNPit v2 and v3. When confronted with the challenge of how to combine probabilistic and logical inference approaches, we tried five different techniques. In the first method, we took the original probabilistic score that did not have the customized algorithm discussed in section 5.4.3 of Chapter 5 and multiply it with the logical score. The second method was to include the customized algorithm in the probabilistic score and multiply it with the logical score. In the third method, we took the customized probabilistic and logical scores, added them together and divided by two. In the fourth method, we applied weights to both measures in the form of .1 and .9 to the customized probabilistic and logical scores and then added them together. In the fifth method, we took a combination of the approaches and first added the customized probabilistic and logical scores together, then subtracted it from the product of the customized probabilistic and logical score (Figure 6.12).

Original probabilistic * logical	UII-Prob * Logi
SNPit custom probabilistic * logical	SNPit-Prob * Logi
Average	(Prob + Logi)/2
Weighted Average	0.1*Prob + 0.9*Logi
Probabilistic Combination	Prob+Logi-(Prob*Logi)

Figure 6.12. Various approaches to combining logical and probabilistic scores.

Level	1	50	100	150
TP	1	49	98	147
FP	0	1	2	3
TN	250	249	248	247
FN	249	201	152	103
	1	50	100	150
Precision	1	0.98	0.98	0.98
Recall / Sensitivity	0.004	0.196	0.392	0.588
Specificity	1	0.996	0.992	0.988
Accuracy	0.502	0.596	0.692	0.788
1-Specificity	0	0.004	0.008	0.012

Table 6.6. Portion of the spreadsheet used to create the ROC curve for SNPit v4 with both probabilistic and logical inference.

A section of the spreadsheet used to record the true positives, false positives, true negatives, and false negatives for four levels of SNPit v4 with probabilistic and logical inference. Graphing sensitivity versus 1-specificity produces a ROC curve with good predictive meaning because it curves towards the upper top left corner of the graph. When we examined the ROC curves of the five different methods that were used to combine the logical and probabilistic scores, the second method, customized probabilistic score multiplied by the logical score performed the best (Figure 6.13).

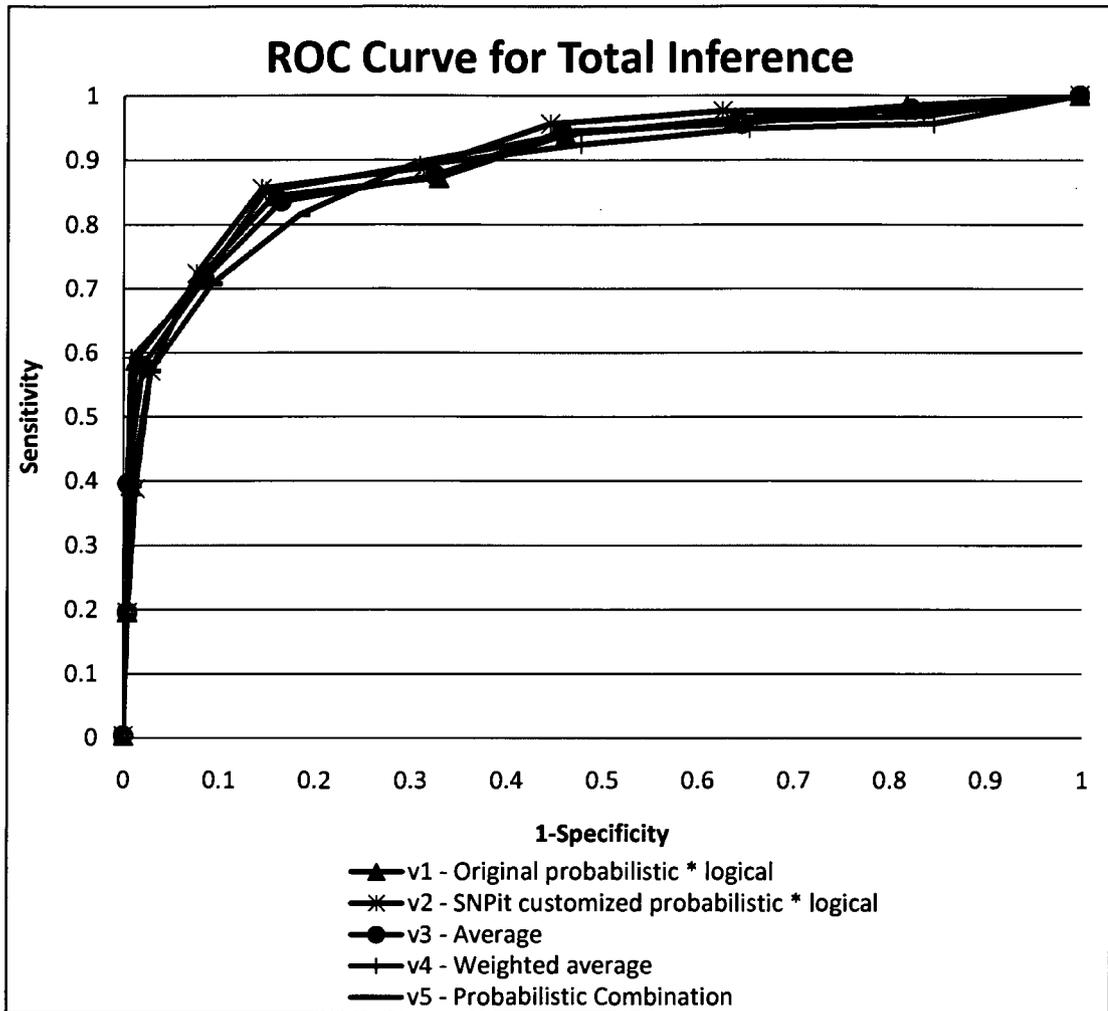


Figure 6.13. ROC curves of the five different methods used to combine the logical and probabilistic scores.

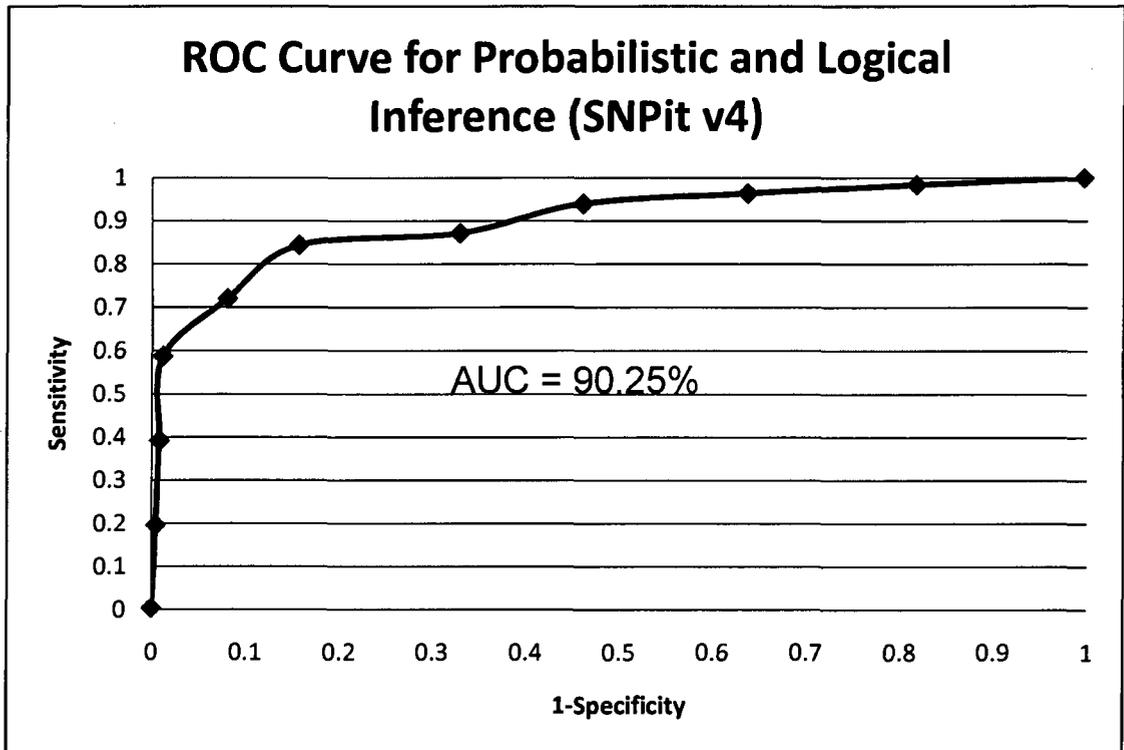


Figure 6.14. ROC curve for SNPit v4 using the method that performed the best, the customized probabilistic score was multiplied with the logical score, graph demonstrates good predictive value as it curves towards the top left corner of the graph, the area under the curve was 90.25%.

The ROC curves of the five methods revealed that the second method, multiplication of the customized probabilistic score with the logical score performed the best (Figure 6.14). The AUC for this ROC curve was 90.25%.

6.4.5 Cross cutting evaluation of SNPit v1-v4

In order to assess the accuracy of versions 1 through 4 of SNPit, with baseline federated integration, logical, probabilistic, and logical with probabilistic

inference, we carried out three forms of evaluation: multiple comparison ROC curve, multiple comparison precision/recall graph, and accuracy calculations. The ROC curves and precision/recall graphs allowed us to cross evaluate the results ranked by probabilistic and logical inference with results ranked by logical inference with results ranked by probabilistic inference with results that are ranked randomly. The random rankings category was created by randomly generating a list of 500 scores that were used as ranking scores, the same procedure as before was carried out, with recall and precision measures taken at 50 level intervals. The accuracy figures allowed us to form the same comparisons as the ROC curves and precision/recall, but we also used it to compare SNPit version 2,3,4 with version 1.

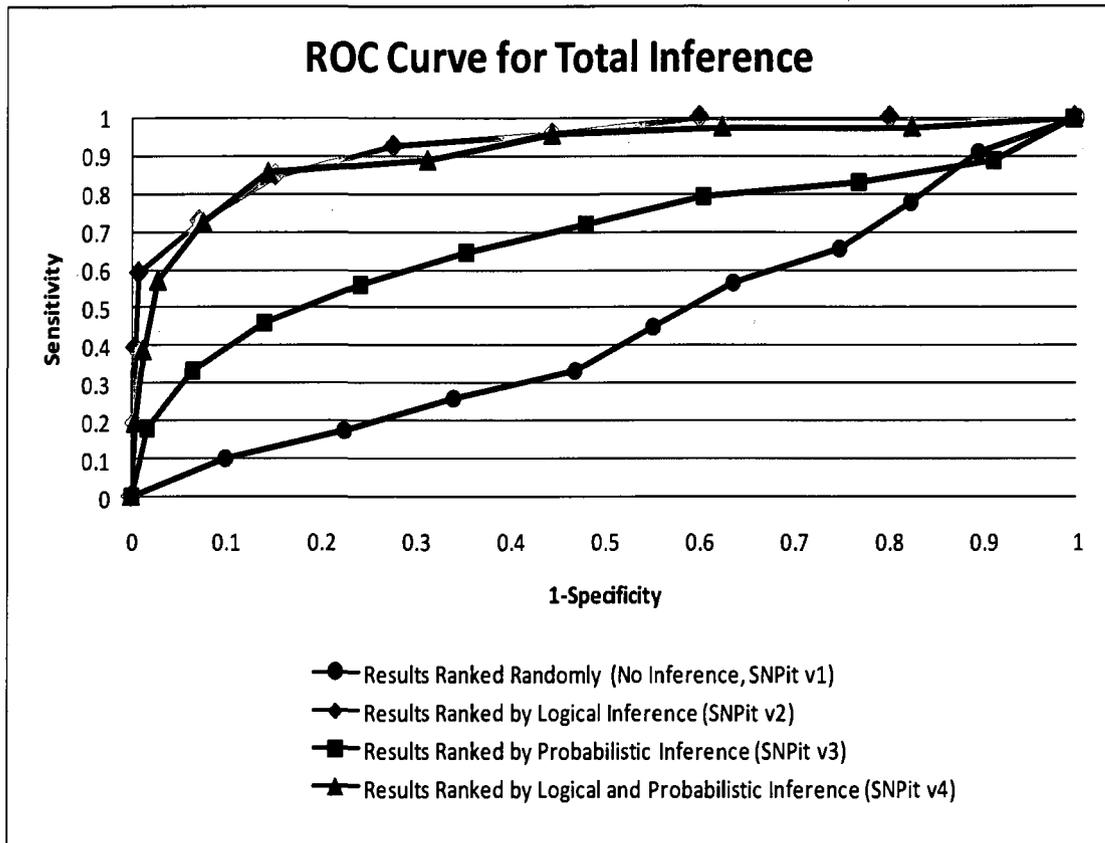


Figure 6.15. ROC curve for multiple comparison between SNPit results ranked by probabilistic inference, results ranked by logical inference, results ranked by probabilistic inference, and results ranked randomly with no inference.

The ROC curve for multiple ranking comparison shows that results ranked by logical inference performed the best out, though results ranked by logical and probabilistic inference performs approximately the same, though slightly lower. Results ranked by probabilistic inference performed moderately well, and results ranked randomly performed as expected (Figure 6.15). The precision/recall graph demonstrates the same result, with results ranked by logical inference performing the

best, with results ranked by logical and probabilistic inference performing the second best (Figure 6.16).

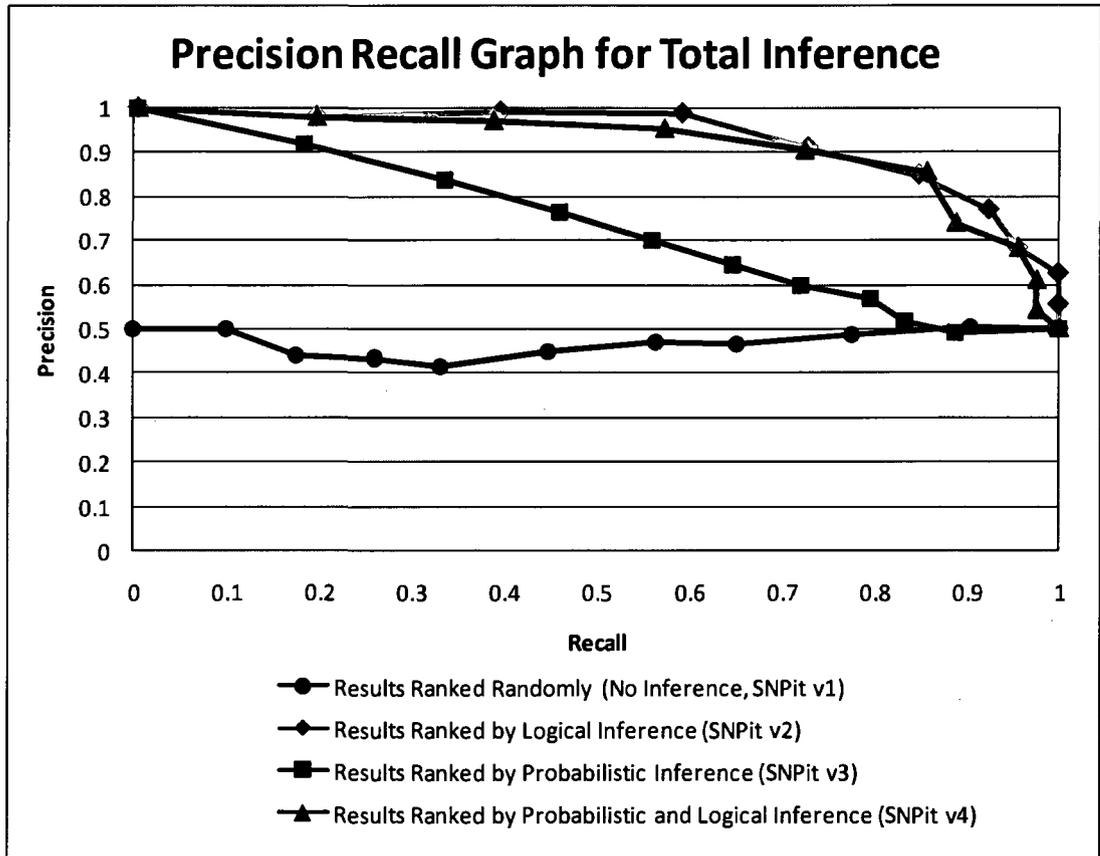


Figure 6.16. Precision/recall curve for multiple comparison between SNPit results ranked by probabilistic inference, results ranked by logical inference, results ranked by probabilistic inference, and results ranked randomly with no inference.

With ranked List		Accuracy
probabilistic		0.648
logical		0.848
prob+log		0.856
random		0.448
No Ranked List		Accuracy
missense/nonsens		0.618543
splicing		0.587304
regulatory		0.554276
Previous domain		Accuracy
logical (Eithon)		0.736
probabilistic (Brent)		0.7

Table 6.7. Accuracy measures for ranked list and no ranked list versions of SNPit, at an optimal threshold of 85% sensitivity.

Finally, in order to evaluate the ranked lists provided by SNPit versions 2,3,4 (logical, probabilistic, and logical combined with probabilistic) with version 1 of SNPit, which does not provide a ranked list but does provide federated integration for different categories, we calculated the accuracy measures. The results of the accuracy calculations demonstrate that logical and logical with probabilistic inference performs the best out of all the different versions of SNPit. At an optimal threshold of 85% sensitivity, probabilistic and logical performs the best, though not by a significant amount. Using the ROC and AUC measures, logical inference performs the best out of all the different versions of SNPit.

When we compared our accuracy measures with that done previously in the domain of gene annotation, we discovered that in our instantiation of logical inference, we performed better than logical inference plus data integration for gene annotation. On the other hand, our instantiation of probabilistic inference did not perform as well when compared to probabilistic inference and data integration applied in the domain of protein annotation. Finally, the discovery that the combination of logical inference and probabilistic inference not adding anything significant to logical inference alone was not an expected results. This paradox could be caused by limitations to our study, discussed in Chapter 7, section 7.3. Or, this could also be an indication that using uncertainty metrics on the data sources and records related to SNP functionality simply does not help for predicting the functional impact of the SNP.

6.5 Measuring the Usability and Usefulness of SNPit, User based evaluation

Qualitative analysis was conducted to incorporate user preferences into the SNPit system. IRB approval was received for the study beforehand; the objective was to conduct a brief open-ended interview to help in the creation of a survey that can help assess subjective utility of the developed SNPit. A focus group of four participants from the University of Washington and the Fred Hutchinson Cancer Research Center were recruited to do a baseline qualitative open ended observational

assessment. The participants will be made up of a combination of graduate students, postdoctoral students, and staff scientists. The average number of years of experience with SNPs was four. The participants were recruited through email and through personal connections with the Fred Hutchinson Cancer Research Center.

We spent 60 minutes with each of the participants conducting both a think aloud procedure and a semi-structured interview to gain some feedback on the SNPit system. The participants were instructed on the general premise of the SNPit system and given the chance to navigate through the web servlet. The think aloud procedure consisted of two SNP annotation tasks to complete using the SNPit tool asking the participants to do a “think-aloud” as they used the tool focusing on both usability and usefulness. The second part of the interviews consisted of three sets of open ended questions aimed at eliciting a) usability, and b) usefulness. The consent form and initial questions used in the semi-structured interview is available in Appendix E.

For the first section of the “thinkaloud” process, the participants were asked to first imagine the scenario: “You’ve just found a SNP and would like some background information on it?”. They were then asked to submit a real world SNP that they were conducting studies on and navigate through the website with that SNP. They were also asked to voice any initial thoughts they might have during this navigation process. For the second annotation task, the participants were asked to imagine the following scenario: “Say you have 5 different SNPs that you are studying, how would you begin rank its potential significance?”. Again, they were

asked to vocalize any thoughts they had during this annotation task. Their responses were then recorded in notes form, making sure that there only a pre-assigned ID was listed on their notes, and not their actual names.

After the two annotation tasks were completed. The participants were asked a series of open ended questions (Appendix E). Their responses was again recorded in notes form with only their assigned ID recorded along with their responses.

The results from this study was then examined in total and various themes and patterns were drawn out from the notes. The feedback from our focus group was positive for all the participants. Overall, all the participants liked the user interface of SNPit in terms of its usability. They liked the basic layout, the colors of the website, as well as the general look and feel. They also felt that the SNPit system would be useful for fellow researchers.

Some themes that emerged from the interviews can be well represented through the following participant quotes:

“This tool is really useful for people with a lot of data and SNP hits that aren’t easily explainable. It can help with explaining our hits and narrowing down interesting SNPs that might need further exploration”.

“I like how straightforward the system is, people can find what they need easily.”

“For a survey, potentially having a combination of Likert and open ended questions would be good.”

Observing the participants during the think aloud process also helped us to identify areas of the website that can be improved. For example, the “Search All Sources” and “Ranking SNPs” link was not easily found by the participants, one participant suggested a pull-down menu to make the organization of the menu selections more understandable. While performing their two annotation tasks, many of the subjects asked for more descriptions of what the results from the individual data sources mean; so a more detailed help section for each of the menu links would be helpful. This would entail adding instructions on each menu link, including explanations for each of the data sources that the menu links extract information from. The need for this level of detailed explanations stems from the fact that the data sources themselves often do not provide adequate information on how to use and understand the information they provide. These changes in the design of the website would be easy to implement and would be related to the future work of this dissertation.

6.6 Discussion

The evaluation of SNPit versions 1-4 resulted in the revelation that logical inference for a ranked list of SNPs performed the best out of all the forms of inference if we look at the ROC curves and AUC. The second best performing inference approach was probabilistic with logical inference combined by using the customized probabilistic score and multiplying it with the logical score. If we use the optimal threshold of 85% sensitivity, then the probabilistic and logical inference version performs better than logical inference alone, though not by any significant amount. The reasons for why logical inference performed the best is likely due to the following factors: 1) only two of the five categories of functional SNP data sources provided preferential information to the SNPs that ended up being counted in our source of true positives (HGMD), as discussed in section 6.4.3; 2) the custom made algorithm that was used to create the SNP score as described in section 5.4.3 of Chapter 5 was not the optimum algorithm that could have been used; and 3) the UII metrics that we used for the SNPit probabilistic inference component was too closely grouped together. Nevertheless, probabilistic inference performed moderately well during the evaluation, and probabilistic with logical inference performed comparably with logical inference alone. A more conservative prediction algorithm would be the probabilistic with logical inference approach.

Chapter 7: Summary and Conclusions

7.1 Introduction

In this dissertation, we created a functional SNP annotation system using a federated data integration system and combinations of logical and probabilistic inference. We then used an iterative design approach to create the four versions of SNPit: SNPit v1 being the baseline system with federated data integration, SNPit v2 adding on logical inference, SNPit v3 adding on probabilistic inference, and SNPit v4 adding on both logical and probabilistic inference. Evaluations using a combination of sensitivity, specificity, recall, and precision were then used for each iteration of the system. A focus group was also used to conduct a preliminary user based evaluation of the system.

7.2 Contributions

The main research question in this dissertation is if it is both feasible and valuable to create and evaluate a functional SNP annotation system using federated data integration as the foundation and building logical, probabilistic and probabilistic

with logical inference over the returned data. The main contributions of this dissertation are as follows:

- The development and implementation of a baseline federated data integration system for the purposes of SNP annotation as described in Chapter 5, section 5.4.1. We additionally tried to include a comprehensive number of data sources related to SNP annotation which was a gap that existed in other SNP annotation systems.
- The development and implementation of logical, probabilistic, and logical combined with probabilistic inference for a SNP annotation system as described in Chapter 5, sections 5.4.2, 5.4.3, and 5.4.4. Currently there are no SNP annotation systems that conduct inference in these areas. The combination of data integration / probabilistic inference / logical inference for informatics in general had not existed to our knowledge prior to the contributions of this dissertation.
- A formal evaluation of a SNP annotation system which had previously never been undertaken to our knowledge. As described in Chapter 6, sections 6.4.1, 6.4.2, 6.4.3, 6.4.4., and 6.4.5., our evaluation found that logical inference performs the best out of all the inference methods that we tried and that probabilistic inference did not significantly add to the predictive ability of logical inference. Our evaluation also added a contribution in identification

of a source that could be used as an alternative gold standard since none exists at present in the SNP annotation domain.

- Our findings on logical, probabilistic, and combinations of the both in the domain of SNP annotation as compared to previous work done in the domain of gene annotation revealed that SNP annotation performs better than gene annotation when using logical inference, but does not perform as well when using probabilistic inference.

7.3 Limitations

The findings of this dissertation are not necessarily generalizable outside the domain of SNP annotation and outside of the specific logical and probabilistic methods we used. Furthermore, there are certain limitations that need to be kept in mind when examining the results.

As previously discussed in Chapter 6, section 6.2.1, we were faced with the challenge of identifying a suitable database that could be used as an alternative gold standard. Even though we felt that our choice of HGMD was a valid alternative standard because the results that are published in that website goes through a manual curation process, there is definitely still possible true negatives that are included in the data set. This number of potential true negatives that exist in HGMD would be hard to estimate. Furthermore, the website that we choose as our source of true negatives (dbSNP) has a 0.008% of possible true positives being included as well.

This can make our evaluations based on these datasets inaccurate, although we estimate that the error rates would be fairly small.

Another category of limitations come from the fact that we did not conduct refinement of neither the decision tree on which the logical inference was built upon nor the UII metrics that the probabilistic inference was built upon. We did not use any optimization techniques to create and design our decision tree, it was based off of SNP expert knowledge and biological literature; thus, there is the potential for variability in the design of the decision tree as well as the heuristic weights assigned to the end nodes on the tree. Furthermore, SNPs that might reside on more than one path was not examined in the decision tree; thus, there is the potential for a SNP to have more than one logical prediction score. On a similar note, the probabilistic inference methods were also not fine tuned. The UII metrics were chosen, again, based on SNP expert knowledge; therefore, the accuracy of these metrics might not be accurate if the expert is simply more proficient at assigning logical metrics as opposed to probabilistic metrics. The probabilistic metrics that were assigned based on expert knowledge could have been too tightly clustered together and thus, wouldn't have been effective at classifying categories. Furthermore, the use of HGMD as our alternative gold standard might have impacted our results if the data weighted certain categories of SNPs more heavily than others. In addition, the customized algorithm as described in section 5.3.4 of Chapter 5 was not tested in any way, and thus, a better customized algorithm could be developed.

Finally, the actual SNPit system itself is limited in its production environment since it is only staffed by one person for maintenance and development purposes. The system is also limited in terms of the speed in which query results are returned. Depending on which sources the user is trying to connect with, queries can take up to a few minutes in order to return information. To decrease the amount of wait time needed, additional development time would be needed.

7.4 Future Work

Improvements to the SNPit system would require fine tuning and modifying both the decision tree used for logical inference as well as the UII uncertainty metrics and customized algorithm used for probabilistic inference. One direction in which such modifications could be made would be to use machine learning to see if our decision tree and customized algorithm could be guided by actual data rather than a heuristic variable approach. Machine learning is defined broadly as a discipline where data sets are used to train an algorithm (180); in informatics, machine learning is often used in the development of Bayesian Networks, Neural Networks, and Hidden Markov Models in order to evaluate the network being created using a training set that includes the predictors and outcomes involved. For the purposes of this dissertation, one could think of machine learning as being used to better optimize the probabilistic uncertainty metrics using sets of true positives and true negatives to train the Pr and Ps values. Whether or not machine learning

would be able to improve the accuracy measurements of our system, though, would not be assured because there is no dataset out there that would have absolute predictors and outcomes due to the lack of a gold standard for both true positives and negatives as first discussed in section 6.2 of Chapter 6.

The existing SNPit system can be extended to explore the impact of common number variants (CNVs) on clinical phenotypes. CNVs, which are segments of hundreds to thousands of DNA sequences that are either inserted or deleted, have been hypothesized as contributing significantly to phenotypic variation (181). Modification of the SNPit system to include information on CNVs would require creating new wrappers to CNVs data and modifying the common data model and user interface to accommodate these new data sources. Since CNVs could include SNPs within them, the relationship between the two would be mapped out using genomic coordinates. Additional logical and probabilistic inference would need to be developed and a new alternative source of true positives and true negatives would need to be found for evaluation purposes. Since CNVs and their role in genome wide association studies are in the beginning stages of research, finding data sources for the purposes of evaluation would be a challenge.

Additional ways in which to improve the results of this study would be to conduct a sensitivity analysis of both the logical and probabilistic metrics to examine whether or not variations in the assignment of these scores would alter the rankings of the results. The random samples from HGMD used in our evaluation of

the probabilistic inference component of the system could include more samples of regulatory and splice site SNPs since they are more likely under-represented in the data source. The evaluation could use actual data sets from real investigators as opposed to the random SNPs from HGMD and dbSNP that we used.

The positive results from our preliminary focus group user evaluation described in section 6.5 of Chapter 6 provided some suggestions concerning improvements to the Menu section of the website and additional instructions that require additional work that needs to be done. All of the different inference methods need to be made available for use on the website with additional information such as the ROC curves should be included. The usage traffic of the website should also be recorded. Furthermore, the preliminary results provided use with some guidelines that can be used to create an online survey that can help us to continue gathering user feedback.

Finally, the numbers of SNPs used during our evaluation could be increased to increase the power of our results. Particularly the evaluation related to versions 2,3,4 and ranked lists, which used a total of 500 SNPs per cycle. This number can be increased in the future evaluations of the system.

7.5 Concluding Remarks

The results of this dissertation were marked with both the expected and unexpected. The feasibility component of the research question: whether we could

build a federated data integration system with combinations of logical and probabilistic inference, was successfully demonstrated. However, we found that probabilistic inference did not contribute significantly to the predictive power of our SNP annotation system, even though it performed very well in a previous domain of gene annotation; whereas the logical inference method performed markedly better than it had when applied to gene annotation. These results, of course, must be taken along with the caveats mentioned in the previous section, but they demonstrate that both logical and probabilistic inference depend on the domain in which it is applied.

Furthermore, since this dissertation is a demonstration of the first time a SNP annotation system has been formally evaluated, the fact that our best performing method produced an area under the curve of greater than 90 percent means that we have demonstrated that informatics techniques can accurately predict which SNPs are likely to be functional. The SNPit system has also been shown to be of great use to researchers who need to further examine the results of SNPs that are found to be statistically significant in an association study.

Bibliography

1. Auffray C, Chen Z, Hood L. Systems medicine: the future of medical genomics and healthcare. *Genome Med* 2009;1(1):2.
2. Hood L, Heath JR, Phelps ME, Lin B. Systems biology and new technologies enable predictive and preventative medicine. *Science* 2004;306(5696):640-3.
3. Zerhouni EA. US biomedical research: basic, translational, and clinical sciences. *Jama* 2005;294(11):1352-8.
4. Snyderman R, Langheier J. Prospective health care: the second transformation of medicine. *Genome Biol* 2006;7(2):104.
5. Health NIo. Policy for sharing of data obtained in NIH supported or conducted genome-wide association studies (GWAS). In: Regist F, editor.; 2007. p. 49290-49297.
6. Thomas D. *Statistical Methods in Genetic Epidemiology*. New York: Oxford University Press; 2004.
7. Attia J, Ioannidis JP, Thakkestian A, McEvoy M, Scott RJ, Minelli C, et al. How to use an article about genetic association: A: Background concepts. *Jama* 2009;301(1):74-81.
8. Hirschhorn JN, Lohmueller K, Byrne E, Hirschhorn K. A comprehensive review of genetic association studies. *Genet Med* 2002;4(2):45-61.
9. Costa L, Eaton D. *Gene-environment interactions: Fundamentals of ecogenetics*. Hoboken, NJ: John Wiley & Sons; 2006.
10. Miniño A HM, Smith B. Deaths: Preliminary Data for 2004 National vital statistics reports. *National Center for Health Statistics* 2006;54(13).
11. Thomas DC. *Statistical Methods in Genetic Epidemiology*. USA: Oxford University Press; 2004.
12. Tsui LC, Buchwald M, Barker D, Braman JC, Knowlton R, Schumm JW, et al. Cystic fibrosis locus defined by a genetically linked polymorphic DNA marker. *Science* 1985;230(4729):1054-7.
13. The Huntington's Disease Collaborative Research Group. A novel gene containing a trinucleotide repeat that is expanded and unstable on Huntington's disease chromosomes. *Cell* 1993;72:971-983.
14. Risch N, Merikangas K. The future of genetic studies of complex human diseases. *Science* 1996;273(5281):1516-7.
15. Kung HC, Hoyert DL, Xu J, Murphy SL. Deaths: final data for 2005. *Natl Vital Stat Rep* 2008;56(10):1-120.
16. Finishing the euchromatic sequence of the human genome. *Nature* 2004;431(7011):931-45.

17. The International HapMap Project. *Nature* 2003;426(6968):789-96.
18. Maresso K, Broeckel U. Genotyping platforms for mass-throughput genotyping with SNPs, including human genome-wide scans. *Adv Genet* 2008;60:107-39.
19. Crawford DC, Nickerson DA. Definition and clinical importance of haplotypes. *Annu Rev Med* 2005;56:303-20.
20. Cordell HJ, Clayton DG. Genetic association studies. *Lancet* 2005;366(9491):1121-31.
21. Pearson TA, Manolio TA. How to interpret a genome-wide association study. *Jama* 2008;299(11):1335-44.
22. Altshuler D, Daly M. Guilt beyond a reasonable doubt. *Nat Genet* 2007;39(7):813-5.
23. Altshuler D, Daly MJ, Lander ES. Genetic mapping in human disease. *Science* 2008;322(5903):881-8.
24. Hindorff LA, Junkins HA, Mehta JP, TA M. A Catalog of Published Genome-Wide Association Studies. [cited; Available from: www.genome.gov/26525384
25. Hindorff LA, Sethupathy P, Junkins HA, Ramos EM, Mehta JP, Collins FS, et al. Potential etiologic and functional implications of genome-wide association loci for human diseases and traits. *Proc Natl Acad Sci U S A* 2009.
26. Swaroop A, Branham KE, Chen W, Abecasis G. Genetic susceptibility to age-related macular degeneration: a paradigm for dissecting complex disease traits. *Hum Mol Genet* 2007;16 Spec No. 2:R174-82.
27. Gudbjartsson DF, Arnar DO, Helgadóttir A, Gretarsdóttir S, Holm H, Sigurdsson A, et al. Variants conferring risk of atrial fibrillation on chromosome 4q25. *Nature* 2007;448(7151):353-7.
28. Moffatt MF, Kabesch M, Liang L, Dixon AL, Strachan D, Heath S, et al. Genetic variants regulating *ORMDL3* expression contribute to the risk of childhood asthma. *Nature* 2007;448(7152):470-3.
29. Baurecht H, Irvine AD, Novak N, Illig T, Buhler B, Ring J, et al. Toward a major risk factor for atopic eczema: meta-analysis of filaggrin polymorphism data. *J Allergy Clin Immunol* 2007;120(6):1406-12.
30. Palmer CN, Ismail T, Lee SP, Terron-Kwiatkowski A, Zhao Y, Liao H, et al. Filaggrin null mutations are associated with increased asthma severity in children and young adults. *J Allergy Clin Immunol* 2007;120(1):64-8.
31. Easton DF, Pooley KA, Dunning AM, Pharoah PD, Thompson D, Ballinger DG, et al. Genome-wide association study identifies novel breast cancer susceptibility loci. *Nature* 2007;447(7148):1087-93.

32. Tomlinson I, Webb E, Carvajal-Carmona L, Broderick P, Kemp Z, Spain S, et al. A genome-wide association scan of tag SNPs identifies a susceptibility variant for colorectal cancer at 8q24.21. *Nat Genet* 2007;39(8):984-8.
33. Zanke BW, Greenwood CM, Rangrej J, Kustra R, Tenesa A, Farrington SM, et al. Genome-wide association scan identifies a colorectal cancer susceptibility locus on chromosome 8q24. *Nat Genet* 2007;39(8):989-94.
34. McPherson R, Pertsemlidis A, Kavaslar N, Stewart A, Roberts R, Cox DR, et al. A common allele on chromosome 9 associated with coronary heart disease. *Science* 2007;316(5830):1488-91.
35. Samani NJ, Erdmann J, Hall AS, Hengstenberg C, Mangino M, Mayer B, et al. Genomewide association analysis of coronary artery disease. *N Engl J Med* 2007;357(5):443-53.
36. The Wellcome Trust Case Control Consortium. Genome-wide association study of 14,000 cases of seven common diseases and 3,000 shared controls. *Nature* 2007;447(7145):661-78.
37. Todd JA, Walker NM, Cooper JD, Smyth DJ, Downes K, Plagnol V, et al. Robust associations of four new chromosome regions from genome-wide analyses of type 1 diabetes. *Nat Genet* 2007;39(7):857-64.
38. Zeggini E, Weedon MN, Lindgren CM, Frayling TM, Elliott KS, Lango H, et al. Replication of genome-wide association signals in UK samples reveals risk loci for type 2 diabetes. *Science* 2007;316(5829):1336-41.
39. Hafler DA, Compston A, Sawcer S, Lander ES, Daly MJ, De Jager PL, et al. Risk alleles for multiple sclerosis identified by a genomewide study. *N Engl J Med* 2007;357(9):851-62.
40. Zheng SL, Sun J, Wiklund F, Smith S, Stattin P, Li G, et al. Cumulative association of five genetic variants with prostate cancer. *N Engl J Med* 2008;358(9):910-9.
41. Winkelmann J, Schormair B, Lichtner P, Ripke S, Xiong L, Jalilzadeh S, et al. Genome-wide association study of restless legs syndrome identifies common variants in three genomic regions. *Nat Genet* 2007;39(8):1000-6.
42. Stefansson H, Rye DB, Hicks A, Petursson H, Ingason A, Thorgeirsson TE, et al. A genetic risk factor for periodic limb movements in sleep. *N Engl J Med* 2007;357(7):639-47.
43. Thomson W, Barton A, Ke X, Eyre S, Hinks A, Bowes J, et al. Rheumatoid arthritis association at 6q23. *Nat Genet* 2007;39(12):1431-3.
44. Graham RR, Kyogoku C, Sigurdsson S, Vlasova IA, Davies LR, Baechler EC, et al. Three functional variants of IFN regulatory factor 5 (IRF5) define risk and protective haplotypes for human lupus. *Proc Natl Acad Sci U S A* 2007;104(16):6758-63.

45. Melzer D, Hogarth S, Liddell K, Ling T, Sanderson S, Zimmern RL. Genetic tests for common diseases: new insights, old concerns. *Bmj* 2008;336(7644):590-3.
46. Kruglyak L, Nickerson DA. Variation is the spice of life. *Nat Genet* 2001;27(3):234-6.
47. Sherry ST, Ward MH, Kholodov M, Baker J, Phan L, Smigielski EM, et al. dbSNP: the NCBI database of genetic variation. *Nucleic Acids Res* 2001;29(1):308-11.
48. dbSNP. dbSNP Data Content Information. 2009 [cited; Available from: <http://www.ncbi.nlm.nih.gov/bookshelf/br.fcgi?book=helpsnpfaq&part=Content>]
49. Yamada R. Primer: SNP-associated studies and what they can teach us. *Nat Clin Pract Rheumatol* 2008;4(4):210-7.
50. McCarthy MI, Abecasis GR, Cardon LR, Goldstein DB, Little J, Ioannidis JP, et al. Genome-wide association studies for complex traits: consensus, uncertainty and challenges. *Nat Rev Genet* 2008;9(5):356-69.
51. Hindorff LA, Junkins HA, Mehta JP, TA M. A Catalog of Published Genome-Wide Association Studies. 3/10/09 [cited; Available from: www.genome.gov/26525384]
52. Feuk L, Carson AR, Scherer SW. Structural variation in the human genome. *Nat Rev Genet* 2006;7(2):85-97.
53. Carlson CS, Eberle MA, Rieder MJ, Yi Q, Kruglyak L, Nickerson DA. Selecting a maximally informative set of single-nucleotide polymorphisms for association analyses using linkage disequilibrium. *Am J Hum Genet* 2004;74(1):106-20.
54. Anderson CA, Pettersson FH, Barrett JC, Zhuang JJ, Ragoussis J, Cardon LR, et al. Evaluating the effects of imputation on the power, coverage, and cost efficiency of genome-wide SNP platforms. *Am J Hum Genet* 2008;83(1):112-9.
55. Burton PR, Tobin MD, Hopper JL. Key concepts in genetic epidemiology. *Lancet* 2005;366(9489):941-51.
56. Pettersson FH, Anderson CA, Clarke GM, Barrett JC, Cardon LR, Morris AP, et al. Marker selection for genetic case-control association studies. *Nat Protoc* 2009;4(5):743-52.
57. Coon KD, Myers AJ, Craig DW, Webster JA, Pearson JV, Lince DH, et al. A high-density whole-genome association study reveals that APOE is the major susceptibility gene for sporadic late-onset Alzheimer's disease. *J Clin Psychiatry* 2007;68(4):613-8.
58. Matarin M, Brown WM, Scholz S, Simon-Sanchez J, Fung HC, Hernandez D, et al. A genome-wide genotyping study in patients with ischaemic stroke: initial analysis and data release. *Lancet Neurol* 2007;6(5):414-20.

59. Perkel J. SNP genotyping: six technologies that keyed a revolution. *Nature Methods* 2008(5):447 - 453.
60. Lamberts SW, Uitterlinden AG. Genetic Testing in Clinical Practice. *Annu Rev Med* 2008.
61. Hunter DJ, Kraft P. Drinking from the fire hose--statistical issues in genomewide association studies. *N Engl J Med* 2007;357(5):436-9.
62. Patterson M, Cardon L. Replication publication. *PLoS Biol* 2005;3(9):e327.
63. Evangelou E, Maraganore DM, Annesi G, Brighina L, Brice A, Elbaz A, et al. Non-replication of association for six polymorphisms from meta-analysis of genome-wide association studies of Parkinson's disease: Large-scale collaborative study. *Am J Med Genet B Neuropsychiatr Genet* 2009.
64. Helgadottir A, Thorleifsson G, Manolescu A, Gretarsdottir S, Blondal T, Jonasdottir A, et al. A common variant on chromosome 9p21 affects the risk of myocardial infarction. *Science* 2007;316(5830):1491-3.
65. Burke W, Khoury MJ, Stewart A, Zimmern RL. The path from genome-based research to population health: development of an international public health genomics network. *Genet Med* 2006;8(7):451-8.
66. Christensen K, Murray JC. What genome-wide association studies can do for medicine. *N Engl J Med* 2007;356(11):1094-7.
67. Hunter DJ, Khoury MJ, Drazen JM. Letting the genome out of the bottle--will we get our wish? *N Engl J Med* 2008;358(2):105-7.
68. Malecki MT, Mlynarski W, Skupien J. Can geneticists help clinicians to understand and treat non-autoimmune diabetes? *Diabetes Res Clin Pract* 2008;82 Suppl 2:S83-93.
69. Crowley JJ, Sullivan PF, McLeod HL. Pharmacogenomic genome-wide association studies: lessons learned thus far. *Pharmacogenomics* 2009;10(2):161-3.
70. Schwarz UI, Ritchie MD, Bradford Y, Li C, Dudek SM, Frye-Anderson A, et al. Genetic determinants of response to warfarin during initial anticoagulation. *N Engl J Med* 2008;358(10):999-1008.
71. Khoury MJ, Gwinn M, Burke W, Bowen S, Zimmern R. Will genomics widen or help heal the schism between medicine and public health? *Am J Prev Med* 2007;33(4):310-7.
72. Khoury MJ, McCabe LL, McCabe ER. Population screening in the age of genomic medicine. *N Engl J Med* 2003;348(1):50-8.
73. Plenge RM, Seielstad M, Padyukov L, Lee AT, Remmers EF, Ding B, et al. TRAF1-C5 as a risk locus for rheumatoid arthritis--a genomewide study. *N Engl J Med* 2007;357(12):1199-209.
74. Primer on Statistical Significance and P Values. *Effective Clinical Practice* July/August 2001 [cited; Available from:

http://www.acponline.org/clinical_information/journals_publications/ecp/julug01/primer.htm

75. Weedon MN, Lettre G, Freathy RM, Lindgren CM, Voight BF, Perry JR, et al. A common variant of HMG2 is associated with adult and childhood height in the general population. *Nat Genet* 2007;39(10):1245-50.
76. Henrikson NB, Bowen D, Burke W. Does genomic risk information motivate people to change their behavior? *Genome Med* 2009;1(4):37.
77. Jakobsdottir J, Gorin MB, Conley YP, Ferrell RE, Weeks DE. Interpretation of genetic association studies: markers with replicated highly significant odds ratios may be poor classifiers. *PLoS Genet* 2009;5(2):e1000337.
78. Khoury MJ, Gwinn M, Yoon PW, Dowling N, Moore CA, Bradley L. The continuum of translation research in genomic medicine: how can we accelerate the appropriate integration of human genome discoveries into health care and disease prevention? *Genet Med* 2007;9(10):665-74.
79. Janssens AC, Pardo MC, Steyerberg EW, van Duijn CM. Revisiting the clinical validity of multiplex genetic testing in complex diseases. *Am J Hum Genet* 2004;74(3):585-8; author reply 588-9.
80. Hanley JA. Receiver operating characteristic (ROC) methodology: the state of the art. *Crit Rev Diagn Imaging* 1989;29(3):307-35.
81. Fleming RE, Bacon BR. Orchestration of iron homeostasis. *N Engl J Med* 2005;352(17):1741-4.
82. Anderson JL, Carlquist JF, Horne BD, Hopkins PN. Progress in unraveling the genetics of coronary artery disease and myocardial infarction. *Curr Atheroscler Rep* 2007;9(3):179-86.
83. Samani NJ, Deloukas P, Erdmann J, Hengstenberg C, Kuulasmaa K, McGinnis R, et al. Large scale association analysis of novel genetic loci for coronary artery disease. *Arterioscler Thromb Vasc Biol* 2009;29(5):774-80.
84. Janssens AC, van Duijn CM. Genome-based prediction of common diseases: methodological considerations for future research. *Genome Med* 2009;1(2):20.
85. Mayeux R, Schupf N. Apolipoprotein E and Alzheimer's disease: the implications of progress in molecular medicine. *Am J Public Health* 1995;85(9):1280-4.
86. Simopoulos AP. Genetic screening: programs, principles, and research--thirty years later. Reviewing the recommendations of the Committee for the Study of Inborn Errors of Metabolism (SIEM). *Public Health Genomics* 2009;12(2):105-11.
87. Workshop on Data Sharing: Governance and Ethics in the CTSA Environment. In: University of Washington; 2009.

88. Caulfield T, McGuire AL, Cho M, Buchanan JA, Burgess MM, Danilczyk U, et al. Research ethics recommendations for whole-genome research: consensus statement. *PLoS Biol* 2008;6(3):e73.
89. Mascalzoni D, Hicks A, Pramstaller P, Wjst M. Informed consent in the genomics era. *PLoS Med* 2008;5(9):e192.
90. Potter BK, Avard D, Entwistle V, Kennedy C, Chakraborty P, McGuire M, et al. Ethical, Legal, and Social Issues in Health Technology Assessment for Prenatal/Preconceptional and Newborn Screening: A Workshop Report. *Public Health Genomics* 2008.
91. Sahai I, Marsden D. Newborn screening. *Crit Rev Clin Lab Sci* 2009;46(2):55-82.
92. Teutsch SM, Bradley LA, Palomaki GE, Haddow JE, Piper M, Calonge N, et al. The Evaluation of Genomic Applications in Practice and Prevention (EGAPP) Initiative: methods of the EGAPP Working Group. *Genet Med* 2009;11(1):3-14.
93. Nakamura Y. DNA variations in human and medical genetics: 25 years of my experience. *J Hum Genet* 2009;54(1):1-8.
94. Higashi MK, Veenstra DL, Kondo LM, Wittkowsky AK, Srinouanprachanh SL, Farin FM, et al. Association between CYP2C9 genetic variants and anticoagulation-related outcomes during warfarin therapy. *Jama* 2002;287(13):1690-8.
95. FDA releases final guidance for pharmacogenomic data. *Pharmacogenomics* 2005;6(3):209.
96. You JH, Tsui KK, Wong RS, Cheng G. Potential Clinical and Economic Outcomes of CYP2C9 and VKORC1 Genotype-Guided Dosing in Patients Starting Warfarin Therapy. *Clin Pharmacol Ther* 2009.
97. Anderson JL, Horne BD, Stevens SM, Grove AS, Barton S, Nicholas ZP, et al. Randomized trial of genotype-guided versus standard warfarin dosing in patients initiating oral anticoagulation. *Circulation* 2007;116(22):2563-70.
98. D'Andrea G, D'Ambrosio RL, Di Perna P, Chetta M, Santacrose R, Brancaccio V, et al. A polymorphism in the VKORC1 gene is associated with an interindividual variability in the dose-anticoagulant effect of warfarin. *Blood* 2005;105(2):645-9.
99. Snow JL, Gibson LE. The role of genetic variation in thiopurine methyltransferase activity and the efficacy and/or side effects of azathioprine therapy in dermatologic patients. *Arch Dermatol* 1995;131(2):193-7.
100. Innocenti F, Undevia SD, Iyer L, Chen PX, Das S, Kocherginsky M, et al. Genetic variants in the UDP-glucuronosyltransferase 1A1 gene predict the risk of severe neutropenia of irinotecan. *J Clin Oncol* 2004;22(8):1382-8.

101. Chung WH, Hung SI, Hong HS, Hsieh MS, Yang LC, Ho HC, et al. Medical genetics: a marker for Stevens-Johnson syndrome. *Nature* 2004;428(6982):486.
102. Mallal S, Nolan D, Witt C, Masel G, Martin AM, Moore C, et al. Association between presence of HLA-B*5701, HLA-DR7, and HLA-DQ3 and hypersensitivity to HIV-1 reverse-transcriptase inhibitor abacavir. *Lancet* 2002;359(9308):727-32.
103. Couzin J, Kaiser J. Genome-wide association. Closing the net on common disease genes. *Science* 2007;316(5826):820-2.
104. Magnus D, Cho MK, Cook-Deegan R. Direct-to-consumer genetic tests: beyond medical regulation? *Genome Med* 2009;1(2):17.
105. 23andMe. [cited; Available from: <https://www.23andme.com/>
106. deCODE. [cited; Available from: <http://www.decode.com/>
107. O'Carroll PW, Yasnoff WA, Ward MH, Ripp LH, Martin EL. *Public Health Informatics and Information Systems*: Springer; 2003.
108. Frayling TM. Genome-wide association studies provide new insights into type 2 diabetes aetiology. *Nat Rev Genet* 2007;8(9):657-62.
109. Lyssenko V, Nagorny CL, Erdos MR, Wierup N, Jonsson A, Spiegel P, et al. Common variant in MTNR1B associated with increased risk of type 2 diabetes and impaired early insulin secretion. *Nat Genet* 2009;41(1):82-8.
110. Pharoah PD, Antoniou AC, Easton DF, Ponder BA. Polygenes, risk prediction, and targeted prevention of breast cancer. *N Engl J Med* 2008;358(26):2796-803.
111. Pohlhaus JR, Cook-Deegan RM. Genomics research: world survey of public funding. *BMC Genomics* 2008;9:472.
112. Hagmann M. Human genome. A good SNP may be hard to find. *Science* 1999;285(5424):21-2.
113. Mooney S. Bioinformatics approaches and resources for single nucleotide polymorphism functional analysis. *Brief Bioinform* 2005;6(1):44-56.
114. Bhatti P, Church DM, Rutter JL, Struwing JP, Sigurdson AJ. Candidate single nucleotide polymorphism selection using publicly available tools: a guide for epidemiologists. *Am J Epidemiol* 2006;164(8):794-804.
115. Ouzounis C, Karp P. The past, present and future of genome-wide re-annotation. *Genome Biology* 2002;3(2):2001.1-2001.6.
116. Tefferi A, Wieben ED, Dewald GW, Whiteman DA, Bernard ME, Spelsberg TC. Primer on medical genomics part II: Background principles and methods in molecular genetics. *Mayo Clin Proc* 2002;77(8):785-808.
117. Alberts B, Bray D, Johnson A, et al. *Essential Cell Biology: An Introduction to the Molecular Biology of the Cell*. New York, NY: Garland Publishing; 1998.

118. Jones PA, Takai D. The role of DNA methylation in mammalian epigenetics. *Science* 2001;293(5532):1068-70.
119. Ben-Dov C, Hartmann B, Lundgren J, Valcarcel J. Genome-wide analysis of alternative pre-mRNA splicing. *J Biol Chem* 2008;283(3):1229-33.
120. Cartegni L, Wang J, Zhu Z, Zhang MQ, Krainer AR. ESEfinder: A web resource to identify exonic splicing enhancers. *Nucleic Acids Res* 2003;31(13):3568-71.
121. Karchin R. Next generation tools for the annotation of human SNPs. *Brief Bioinform* 2009;10(1):35-52.
122. Yuan HY, Chiou JJ, Tseng WH, Liu CH, Liu CK, Lin YJ, et al. FASTSNP: an always up-to-date and extendable service for SNP function analysis and prioritization. *Nucleic Acids Res* 2006;34(Web Server issue):W635-41.
123. Lee PH, Shatkay H. F-SNP: computationally predicted functional SNPs for disease association studies. *Nucleic Acids Res* 2008;36(Database issue):D820-4.
124. Karchin R, Diekhans M, Kelly L, Thomas DJ, Pieper U, Eswar N, et al. LS-SNP: large-scale annotation of coding non-synonymous SNPs based on multiple information sources. *Bioinformatics* 2005;21(12):2814-20.
125. Dantzer J, Moad C, Heiland R, Mooney S. MutDB services: interactive structural analysis of mutation data. *Nucleic Acids Res* 2005;33(Web Server issue):W311-4.
126. Jegga AG, Gowrisankar S, Chen J, Aronow BJ. PolyDoms: a whole genome database for the identification of non-synonymous coding SNPs with the potential to impact disease. *Nucleic Acids Res* 2007;35(Database issue):D700-6.
127. Conde L, Vaquerizas JM, Dopazo H, Arbiza L, Reumers J, Rousseau F, et al. PupaSuite: finding functional single nucleotide polymorphisms for large-scale genotyping purposes. *Nucleic Acids Res* 2006;34(Web Server issue):W621-5.
128. Yue P, Melamud E, Moulton J. SNPs3D: candidate gene and SNP selection for association studies. *BMC Bioinformatics* 2006;7:166.
129. Xu H, Fau - Gregory SG, Gregory Sg Fau - Hauser ER, Hauser Er Fau - Stenger JE, Stenger Je Fau - Pericak-Vance MA, Pericak-Vance Ma Fau - Vance JM, Vance Jm Fau - Zuchner S, et al. SNPselector: a web tool for selecting SNPs for genetic association studies. (1367-4803 (Print)).
130. Mathew CG. New links to the pathogenesis of Crohn disease provided by genome-wide association scans. *Nat Rev Genet* 2008;9(1):9-14.
131. Witte JS. Multiple prostate cancer risk variants on 8q24. *Nat Genet* 2007;39(5):579-80.

132. Cargill M, Altshuler D, Ireland J, Sklar P, Ardlie K, Patil N, et al. Characterization of single-nucleotide polymorphisms in coding regions of human genes. *Nat Genet* 1999;22(3):231-8.
133. Levy A. *Logic-based techniques in data integration*. Norwell, MA, USA: Kluwer Academic Publishers; 2000.
134. Karasawas KA, Baldock R, Burger A. Bioinformatics integration and agent technology. *Journal of Biomedical Informatics* 2004;37(3):205-219.
135. Louie B, Mork P, Martin-Sanchez F, Halevy A, Tarczy-Hornoch P. Data integration and genomic medicine. *J Biomed Inform* 2007;40(1):5-16.
136. Sujansky. W. Heterogeneous database integration in biomedicine. *J Biomed Inform* 2001;34(4):285–298.
137. Musen M, Crubézy M, Ferguson R, Noy NF, Tu S, Vendetti J. *Protégé-2000*. [Software project] 2003 April 9 [cited; Available from: <http://protege.stanford.edu/>]
138. Wikipedia. The free encyclopedia. User interface. [cited 2009 April 19]; Available from: http://en.wikipedia.org/wiki/User_interface
139. Shaker R, Mork P, Brockenbrough J, Donelson L, Tarczy-Hornoch P. The BioMediator System as a Tool for Integrating Databases on the Web. In: *Proceedings of the Workshop on Information Integration on the Web; 2004 August, 2004; Toronto, ON; 2004*.
140. Mork P, Halevy AY, Tarczy-Hornoch P. A Model for Data Integration Systems of BioMedical Data Applied to Online Genetic Databases. In: *Proceedings of the American Medical Informatics Annual Fall Symposium; 2001 Nov. 3-7; Washington, D.C.; 2001*. p. 473-77.
141. Shapiro A. TouchGraph: open source software for graph visualization using spring-layout and focus+context techniques. [Software] 2006 [cited; Available from: <http://www.touchgraph.com/>]
142. Friedman-Hill E. *Jess (Java Expert Systems Shell)*. 2008 [cited; Available from: <http://www.jessrules.com/jess/index.shtml>]
143. Cadag E, Louie B, Myler PJ, Tarczy-Hornoch P. Biomediator data integration and inference for functional annotation of anonymous sequences. *Pac Symp Biocomput* 2007:343-54.
144. Friedman-Hill E. *Jess In Action: Rule-Based Systems in Java*. 1st ed. CT: Manning Publications Co.; 2003.
145. Laun W. *Predicate Calculus and Jess*. Vienna, Austria: Thales Rail Signalling GesmbH; 2009 5 February.
146. Cadag E. *High-Throughput Inference-Supported Protein Characterization Utilizing the BioMediator Data Integration Platform [MS Thesis]*: University of Washington; 2006.
147. Louie B, Detwiler T, Dalvi N, Shaker R, Tarczy-Hornoch P, Suci D. *Incorporating Uncertainty Metrics into a General-Purpose Data Integration*

- System. In: presented at 19th International Conference on Scientific and Statistical Database Management (SSDBM); 2007; Banff, Canada; 2007.
148. Colbourn C. *The Combinatorics of Network Reliability*. New York, NY, USA: Oxford University Press, Inc.; 1987.
 149. Detwiler L., Gatterbauer W., Louie B., Suci D., P. T-H. Integrating and Ranking Uncertain Scientific Data. In: *Proceedings of 25th International Conference on Data Engineering (ICDE)*; 2009; 2009. p. 1235-1238, IEEE.
 150. Louie B. *Modeling Uncertainty in Data Integration for Improving Protein Function Assignment [PhD Dissertation]*; University of Washington; 2008.
 151. Louie B, Detwiler T, Dalvi N, Shaker R, Tarczy-Hornoch P, Suci D. Incorporating Uncertainty Metrics into a General-Purpose Data Integration System. In: presented at 19th International Conference on Scientific and Statistical Database Management (SSDBM); 2007; Banff, Canada; 2007.
 152. Shen TH, Carlson CS, Tarczy-Hornoch P. SNPit: A federated data integration system for the purpose of functional SNP annotation. *Comput Methods Programs Biomed* 2009.
 153. Maglott D, Ostell J, Pruitt KD, Tatusova T. Entrez Gene: gene-centered information at NCBI. *Nucleic Acids Res* 2005;33(Database issue):D54-8.
 154. Stenson PD, Ball EV, Mort M, Phillips AD, Shiel JA, Thomas NS, et al. *Human Gene Mutation Database (HGMD): 2003 update*. *Hum Mutat* 2003;21(6):577-81.
 155. Voight BF, Kudaravalli S, Wen X, Pritchard JK. A map of recent positive selection in the human genome. *PLoS Biol* 2006;4(3):e72.
 156. SeattleSNPs. NHLBI Program for Genomic Applications, SeattleSNPs. [cited 2008 09]; Available from: <http://pga.gs.washington.edu>
 157. Ng PC, Henikoff S. Predicting deleterious amino acid substitutions. *Genome Res* 2001;11(5):863-74.
 158. Karolchik D, Kuhn R, Baertsch R, Barber GP, Clawson H, Diekhans M, et al. The UCSC Genome Browser Database: Vertebrate Multiz Alignment & PhastCons Conservation. In: *Nucleic Acids Res*; 2008 Jan. p. D773-9.
 159. Karolchik D, Kuhn R, Baertsch R, Barber GP, Clawson H, Diekhans M, et al. The UCSC Genome Browser Database: Genscan Gene Predictions. In: *Nucleic Acids Res*; 2008 Jan. p. D773-9.
 160. Karolchik D, Kuhn R, Baertsch R, Barber GP, Clawson H, Diekhans M, et al. The UCSC Genome Browser Database: GNF Expression Atlas 2. In: *Nucleic Acids Res*; 2008 Jan. p. D773-9.
 161. Heinemeyer T, Wingender E, Reuter I, Hermjakob H, Kel AE, Kel OV, et al. Databases on transcriptional regulation: TRANSFAC, TRRD and COMPEL. *Nucleic Acids Res* 1998;26(1):362-7.
 162. Reese MG, Eeckman FH, Kulp D, Haussler D. Improved splice site detection in Genie. *J Comput Biol* 1997;4(3):311-23.

163. NCBI. The NCBI Handbook, The Single Nucleotide Polymorphism Database (dbSNP) of Nucleotide Sequence Variation. 2009 [cited; Available from: <http://www.ncbi.nlm.nih.gov/books/bv.fcgi?rid=handbook.section.ch5.ch5-s1>
164. Stenson PD, Mort M, Ball EV, Howells K, Phillips AD, Thomas NS, et al. The Human Gene Mutation Database: 2008 update. *Genome Med* 2009;1(1):13.
165. Sabeti PC, Reich DE, Higgins JM, Levine HZ, Richter DJ, Schaffner SF, et al. Detecting recent positive selection in the human genome from haplotype structure. *Nature* 2002;419(6909):832-7.
166. Hartl DL, Clark AG. *Principles of Population Genetics*. second edition ed. Sunderland, Massachusetts: Sinauer Associates, Inc.; 1989.
167. Ng PC, Henikoff S. SIFT: Predicting amino acid changes that affect protein function. *Nucleic Acids Res* 2003;31(13):3812-4.
168. Siepel A, Bejerano G, Pedersen JS, Hinrichs AS, Hou M, Rosenbloom K, et al. Evolutionarily conserved elements in vertebrate, insect, worm, and yeast genomes. *Genome Res* 2005;15(8):1034-50.
169. Yeh RF, Lim LP, Burge CB. Computational inference of homologous gene structures in the human genome. *Genome Res* 2001;11(5):803-16.
170. Su AI, Wiltshire T, Batalov S, Lapp H, Ching KA, Block D, et al. A gene atlas of the mouse and human protein-encoding transcriptomes. *Proc Natl Acad Sci U S A* 2004;101(16):6062-7.
171. Akiyama Y. TFSEARCH: Searching Transcription Factor Binding Sites. [cited; Available from: <http://www.rwcp.or.jp/papia/>
172. Reese MG. Berkeley Drosophila Genome Project, BPDG. [cited; Available from: http://www.fruitfly.org/seq_tools/spliceHelp.html
173. Tabor HK, Risch NJ, Myers RM. Candidate-gene approaches for studying complex genetic traits: practical considerations. *Nat Rev Genet* 2002;3(5):391-7.
174. Mailman MD, Feolo M, Jin Y, Kimura M, Tryka K, Bagoutdinov R, et al. The NCBI dbGaP database of genotypes and phenotypes. *Nat Genet* 2007;39(10):1181-6.
175. Shortliffe E, Cimino J. *Biomedical Informatics: Computer Applications in Health Care and Biomedicine*. 3rd ed: Springer; 2006.
176. Eisner R. Basic Evaluation Measures for Classifier Performance. [cited; Available from: <http://www.cs.ualberta.ca/~eisner/measures.html>
177. Shen TH, Carlson CS, Tarczy-Hornoch P. Evaluating the Accuracy of SNPit: a Functional SNP Annotation System. In: *AMIA Translational Summit*; 2009; CA; 2009.
178. Friedman CP, Wyatt JC. *Evaluation Methods in Biomedical Informatics*. 2nd ed: Springer; 2005.

179. Lasko TA, Bhagwat JG, Zou KH, Ohno-Machado L. The use of receiver operating characteristic curves in biomedical informatics. *J Biomed Inform* 2005;38(5):404-15.
180. Wikipedia. The free encyclopedia. Machine learning. [cited 2009 July 9]; Available from: http://en.wikipedia.org/wiki/Machine_learning
181. McCarroll SA. Extending genome-wide association studies to copy-number variation. *Hum Mol Genet* 2008;17(R2):R135-42.

Appendix A: Wrappers

```
/*
 *
 * Created on Oct 11, 2007
 *
 *
 * @author Terry Shen
 *
 *
 * TODO To change the template for this generated type comment go to
 * Window - Preferences - Java - Code Style - Code Templates
 *
 */

package org.biomediator.wrappers;

import javax.servlet.http.Cookie;

import org.biomediator.util.AttribOption;
import org.biomediator.util.StringUtility;

import java.net.URL;

import java.lang.Object.*;
import java.lang.Class.*;

import java.io.*;
import java.net.*;
import java.util.*;
import java.util.regex.Matcher;
import java.util.regex.Pattern;

public class BDGPWrapper extends AbstractWrapper {
```

```

public static final String TAG_WRAPPER_SOURCE           =
"BDGP_source";
public static final String HELP_TEXT                   = "
Here's some help stuff... ";

public static final String BDGP_HELP_TEXT =
    "    Description:\n" +
    "        Provides an interface for searching the BDGP database.\n"
+
    "\n" +
    "    Usage:\n" +
    "    Servlet examples:\n" +
    "    Usage:\n" +
    "    Servlet examples:\n" +
    "
?term=term[tvalue=aataatagctgtttctctgtgtttaaggcactacaaatactgtggcagcatataat
tcccaggtggccggcgcttcaggtgagtggcaccagcccctggaagcccgg,hexencoded=false,fi
eld=text]";

//rs2076530
//GGCCAGTTTGGATCTGAAGGTGGTAAGTAAGAATTCTAGATAGA
TATTTTG
//http://www.fruitfly.org/cgi-
bin/seq_tools/splice.pl?organism=human&which=both&reverse=no&min_do
nor=0.4&min_acc=0.4&text=aataatagctgtttctctgtgtttaaggcactacaaatactgtggc
agcatataattcccaggtggccggcgcttcaggtgagtggcaccagcccctggaagcccgg

public static final String BDGP_SOURCE_TAG = "BDGP_source";
public static final String BDGP_CMD_TERM = "term";
public static final String BDGP_OPTION_MAP_FIELD = "field";
public static final String BDGP_OPTION_MAP_TVALUE = "tvalue";
public static final String BDGP_OPTION_MAP_HEXENCODED =
"hexencoded";

public static final String CDATA_START           = "<![CDATA[ ";
public static final String CDATA_END           = " ]]>";
public static final String ENDLINE             = " ";

```

```
public static final String urlBaseXML = "http://fruitfly.org:9005/cgi-
    bin/seq_tools/splice.pl?organism=human&which=both&reverse=no&min_do
    nor=0.4&min_acc=0.4&";
```

```
//(?: ... ) non capture group
```

```
public static final String prePatternBDGP = "<html>(.*?)</html>";
public static final String BDGPDonorPattern1 = "<h3>Donor site predictions
for .*<\h3><pre><b>Start End Score Exon Intron<\b>(.*?)<hr>";
public static final String BDGPDonorPattern2 =
"<br>\s+(\d+)\s+(\d+)\s+(\d[.]\d+)\s+(\w+)<font
size=\"\|+2\">(\w)</font><font size=\"\|+2\">(\w)</font>(\w+)";
```

```
public static final String BDGPAcceptorPattern1 =
"<pre></pre><p><h3>Acceptor site predictions for .*<\h3><pre><b>Start
End Score Intron Exon<\b>(.*?)<hr>";
public static final String BDGPAcceptorPattern2 =
"<br>\s+(\d+)\s+(\d+)\s+(\d[.]\d+)\s+(\w+)<font
size=\"\|+2\">(\w)</font><font size=\"\|+2\">(\w)</font>(\w+)";
```

```
public String proxyUrl = null;
private Pattern pattern = null;
private Matcher matcher = null;
private HashMap queryMap = null;
```

```
private String ttype = null;
private String tvalue = null;
private String field = null;
private String hexencoded = null;
```

```
private String convertedQuery = null;
private String retQuery = null;
```

```
public static void main(String[] args) {
```

```
    BDGPWrapper BDGP = new BDGPWrapper();           //invoking
    the constructor
    BDGP.out = new PrintWriter(System.out, true);
```

```

if (args.length > 0) {
    BDGP.processRequest(args[0]);
}
else {
    BDGP.processRequest("");
}
}

public void processRequest(String query) {

    // Make sure the servlet is initialized.
    initializeServlet(BDGP_HELP_TEXT, BDGP_SOURCE_TAG,
        getWrapperProperties("BDGP"));

    // Parse the query string and process commands.
    if (query == null || query.equals("")) {
        query = CMD_HELP;
    }
   _ATTRIB_OPTION ao = new_ATTRIB_OPTION(query);
    processCommands(ao, getWrapperProperties("bdgp"));

    // Check to see if we should exit immediately.
    if (isExitCmd) {
        return;
    }

    String proxy =
        wrapperProperties.getProperty(WrapperProperties.PROP_PROXY_URL);
    if (proxy == null) {
        proxyUrl = "";
    }
    else {
        proxyUrl = proxy + "?";
    }

    TreeSet termTypeSet = ao.getAttribMapSet(BDGP_CMD_TERM);

    if (termTypeSet != null) {

        convertedQuery = constructQuery(termTypeSet.iterator());
        //out.println("The convertedquery is: " + convertedQuery);
    }
}

```

```

        retQuery = field + "=" + convertedQuery ;
        //out.println("The retQuery is: " + retQuery);
    }

try {

    // Download Original XML page.
    String queryURLXML = proxyUrl + urlBaseXML + retQuery;
    BufferedReader brXML = new BufferedReader(new InputStreamReader(new
    URL(queryURLXML).openStream()));
    StringBuffer lineBufXML = new StringBuffer();
    String lineXML = null;

    //out.println("The returned XML output URL is: " + queryURLXML);

    while ((lineXML = brXML.readLine()) != null) {
        lineBufXML.append(lineXML);
    }

    String dataXML = lineBufXML.toString();
    //out.println("This is the dataXML: "+ dataXML);

    out.println(HEADER_XML);

    printElement("bdgp_source",0);

    out.println("\t<bdgp_id>" + CDATA_START + convertedQuery +
    CDATA_END + "</bdgp_id>");

    pattern = Pattern.compile(BDGPDonorPattern1);
    matcher = pattern.matcher(dataXML);
    //out.println(matcher);
    //out.println(pattern);
    if (matcher.find()) { // results are complete
        String dataRows = matcher.group(1);
        //System.err.println("group 1 = "+matcher.group(1));
        pattern = Pattern.compile(BDGPDonorPattern2);
        matcher = pattern.matcher(dataRows);
        while(matcher.find()) {

```

```

        printElement("splicesite",0);
        out.println("\t<bdgp_id>" + CDATA_START +
convertedQuery + CDATA_END + "</bdgp_id>");
        printElement("bdgp_donor_start", matcher.group(1), 1);
        printElement("bdgp_donor_end", matcher.group(2), 1);
        printElement("bdgp_donor_score", matcher.group(3), 1);
        printElement("bdgp_donor_exon", matcher.group(4), 1);
        printElement("bdgp_donor_exon_end", matcher.group(5), 1);
        printElement("bdgp_donor_intron_start", matcher.group(6),
1);
        printElement("bdgp_donor_intron", matcher.group(7), 1);
        printElement("/splicesite",0);
    } //end of while loop

} //end of first big if loop

pattern = Pattern.compile(BDGP AcceptorPattern1);
matcher = pattern.matcher(dataXML);
if (matcher.find()) { // results are complete
    String dataRows = matcher.group(1);
    //System.err.println("group 1 = "+matcher.group(1));
    pattern = Pattern.compile(BDGP AcceptorPattern2);
    matcher = pattern.matcher(dataRows);
    while(matcher.find()) {
        printElement("splicesite",0);
        out.println("\t<bdgp_id>" + CDATA_START +
convertedQuery + CDATA_END + "</bdgp_id>");
        printElement("bdgp_acceptor_start", matcher.group(1), 1);
        printElement("bdgp_acceptor_end", matcher.group(2), 1);
        printElement("bdgp_acceptor_score", matcher.group(3), 1);
        printElement("bdgp_acceptor_intron", matcher.group(4), 1);
        printElement("bdgp_acceptor_intron_end", matcher.group(5),
1);
        printElement("bdgp_acceptor_exon_start", matcher.group(6),
1);
        printElement("bdgp_acceptor_exon", matcher.group(7), 1);
        printElement("/splicesite",0);
    } //end of while loop
} //end of second if loop

```

```

        printElement("/bdgp_source",0);

    brXML.close();

} //end of try statement

catch (Exception e) {

    out.println(HEADER_XML);
    System.err.println(e.getClass() + ": " + e.getMessage());
    e.printStackTrace();

} //end of catch

} //end of processRequest

public String constructQuery(Iterator queryIterator) {

    StringBuffer termBuf = new StringBuffer("");
    HashMap groupMap = new HashMap();

    String retID = null;

    //out.println("This is the queryIterator: " + queryIterator);

    if (queryIterator.hasNext()) {
        while (queryIterator.hasNext()) {
            // Retrieve the term values that were parsed from the query URL.
            HashMap queryMap = (HashMap) queryIterator.next();
            ttype = (String) queryMap.get(AttribOption.OPTION_PARAM_NAME);
            tvalue = (String) queryMap.get(BDGP_OPTION_MAP_TVALUE);
            field = (String) queryMap.get(BDGP_OPTION_MAP_FIELD);
            hexencoded = (String)
            queryMap.get(BDGP_OPTION_MAP_HEXENCODED);

            //out.println("String retID is: " + retID);

            // Convert term values from hex to ascii if necessary.

```

```

        if (hexencoded != null && hexencoded.equals("true")) {
            tvalue = StringUtility.convertHexToASCII(tvalue);

            retID = tvalue;
        }

        else {

            retID = tvalue;
        }

    }
}

//out.println("This is the retID value: " + retID);

return retID;
}

```

```

public String getUsage() {
    return(
        HEADER_XML + "\n" +
        "<" + TAG_WRAPPER_SOURCE + ">\n" +
        " <help>\n" +
        " <usage>\n" +
        " <![CDATA[ \n" + HELP_TEXT +
        " ]]>\n" +
        " </usage>\n" +
        " </help>\n" +
        "</" + TAG_WRAPPER_SOURCE + ">"
    );
}

```

```

public void printElement(String tag, String content, int tabs) {
    String tabStr = "";
    content = content.trim();
    for(int o = 0; o < tabs; o++)
        tabStr += "\t";
}

```

```
        out.println(tabStr + "<" + tag + ">" + CDATA_START + content +
            CDATA_END + "</" + tag + ">");
    }

    public void printElement(String tag, int tabs) {
        String tabStr = "";
        for(int o = 0; o < tabs; o++)
            tabStr += "\t";
        out.println(tabStr + "<" + tag + ">");
    }

    public void terminate() {
        // TODO Auto-generated method stub
    }
}

/*
 *
 * Created on Oct 11, 2007
 *
 *
 * @author Terry Shen
 *
 *
 * TODO To change the template for this generated type comment go to
 * Window - Preferences - Java - Code Style - Code Templates
 *
 *
 */

package org.biomediator.wrappers;

import javax.servlet.http.Cookie;

import org.biomediator.util.AttribOption;
```

```

import org.biomediator.util.StringUtility;

import java.net.URL;

import java.lang.Object.*;
import java.lang.Class.*;

import java.io.*;
import java.net.*;
import java.util.*;
import java.util.regex.Matcher;
import java.util.regex.Pattern;

public class BDGPWrapper extends AbstractWrapper {

    public static final String TAG_WRAPPER_SOURCE =
        "BDGP_source";
    public static final String HELP_TEXT =
        "Here's some help stuff... ";

    public static final String BDGP_HELP_TEXT =
        "    Description:\n" +
        "        Provides an interface for searching the BDGP database.\n"
+
        "\n" +
        "    Usage:\n" +
        "    Servlet examples:\n" +
        "    Usage:\n" +
        "    Servlet examples:\n" +
        "

?term=term[tvalue=aataatagctgttctctgttgtttaaggcactacaaactgtggcagcatataat
tcccaggtggccggccttcaggtgagtgaccagcccctggaagcccgg,hexencoded=false,fi
eld=text]";

//rs2076530
//GGCCAGTTTGGATCTGAAGGTGGTAAGTAAGAATTCTAGATAGA
TATTTTG

```

```
//http://www.fruitfly.org/cgi-
bin/seq_tools/splice.pl?organism=human&which=both&reverse=no&min_do
nor=0.4&min_acc=0.4&text=aataatagctgtttctctgtttaaaggcactacaaatactgtggc
agcatataatttcccaggtggccggcgcttcaggtgagtgccaccagccccctggaagcccg
```

```
public static final String BDGP_SOURCE_TAG = "BDGP_source";
public static final String BDGP_CMD_TERM = "term";
public static final String BDGP_OPTION_MAP_FIELD = "field";
public static final String BDGP_OPTION_MAP_TVALUE = "tvalue";
public static final String BDGP_OPTION_MAP_HEXENCODED =
"hexencoded";
```

```
public static final String CDATA_START      = "<![CDATA[ ";
public static final String CDATA_END        = " ]]>";
public static final String ENDLINE          = " ";
```

```
public static final String urlBaseXML = "http://fruitfly.org:9005/cgi-
bin/seq_tools/splice.pl?organism=human&which=both&reverse=no&min_do
nor=0.4&min_acc=0.4&";
```

```
//(?: ... ) non capture group
```

```
public static final String prePatternBDGP = "<html>(.)</html>";
public static final String BDGPDonorPattern1 = "<h3>Donor site predictions
for .*</h3><pre><b>Start End Score Exon Intron</b>(.*?)<hr>";
public static final String BDGPDonorPattern2 =
"<br>\s+(\d+)\s+(\d+)\s+(\d[.]\d+)\s+(\w+)<font
size=\"\|+2\">(\w)</font><font size=\"\|+2\">(\w)</font>(\w+)";
```

```
public static final String BDGPAcceptorPattern1 =
"<pre></pre><p><h3>Acceptor site predictions for .*</h3><pre><b>Start
End Score Intron Exon</b>(.*?)<hr>";
public static final String BDGPAcceptorPattern2 =
"<br>\s+(\d+)\s+(\d+)\s+(\d[.]\d+)\s+(\w+)<font
size=\"\|+2\">(\w)</font><font size=\"\|+2\">(\w)</font>(\w+)";
```

```
public String proxyUrl = null;
private Pattern pattern = null;
private Matcher matcher = null;
private HashMap queryMap = null;
```

```

private String ttype = null;
private String tvalue = null;
private String field = null;
private String hexencoded = null;

private String convertedQuery = null;
private String retQuery = null;

public static void main(String[] args) {

    BDGPWrapper BDGP = new BDGPWrapper();           //invoking
        the constructor
    BDGP.out = new PrintWriter(System.out, true);
    if (args.length > 0) {
        BDGP.processRequest(args[0]);
    }
    else {
        BDGP.processRequest("");
    }
}

public void processRequest(String query) {

    // Make sure the servlet is initialized.
    initializeServlet(BDGP_HELP_TEXT, BDGP_SOURCE_TAG,
        getWrapperProperties("BDGP"));

    // Parse the query string and process commands.
    if (query == null || query.equals("")) {
        query = CMD_HELP;
    }
    Attribution ao = new Attribution(query);
    processCommands(ao, getWrapperProperties("bdgp"));

    // Check to see if we should exit immediately.
    if (isExitCmd) {
        return;
    }
}

```

```

String proxy =
    wrapperProperties.getProperty(WrapperProperties.PROP_PROXY_URL);
if (proxy == null) {
    proxyUrl = "";
}
else {
    proxyUrl = proxy + "?";
}

TreeSet termTypeSet = ao.getAttribMapSet(BDGP_CMD_TERM);

if (termTypeSet != null) {

    convertedQuery = constructQuery(termTypeSet.iterator());
    //out.println("The convertedquery is: " + convertedQuery);

    retQuery = field + "=" + convertedQuery ;
    //out.println("The retQuery is: " + retQuery);
}

try {

    // Download Original XML page.
    String queryURLXML = proxyUrl + urlBaseXML + retQuery;
    BufferedReader brXML = new BufferedReader(new InputStreamReader(new
    URL(queryURLXML).openStream()));
    StringBuffer lineBufXML = new StringBuffer();
    String lineXML = null;

    //out.println("The returned XML output URL is: " + queryURLXML);

    while ((lineXML = brXML.readLine()) != null) {
        lineBufXML.append(lineXML);
    }

    String dataXML = lineBufXML.toString();
    //out.println("This is the dataXML: "+ dataXML);

    out.println(HEADER_XML);
}

```

```

printElement("bdgp_source",0);

out.println("\t<bdgp_id>" + CDATA_START + convertedQuery +
CDATA_END + "</bdgp_id>");

pattern = Pattern.compile(BDGPDonorPattern1);
matcher = pattern.matcher(dataXML);
//out.println(matcher);
//out.println(pattern);
if (matcher.find()) { // results are complete
    String dataRows = matcher.group(1);
    //System.err.println("group 1 = "+matcher.group(1));
    pattern = Pattern.compile(BDGPDonorPattern2);
    matcher = pattern.matcher(dataRows);
    while(matcher.find()) {
        printElement("splicesite",0);
        out.println("\t<bdgp_id>" + CDATA_START +
convertedQuery + CDATA_END + "</bdgp_id>");
        printElement("bdgp_donor_start", matcher.group(1), 1);
        printElement("bdgp_donor_end", matcher.group(2), 1);
        printElement("bdgp_donor_score", matcher.group(3), 1);
        printElement("bdgp_donor_exon", matcher.group(4), 1);
        printElement("bdgp_donor_exon_end", matcher.group(5), 1);
        printElement("bdgp_donor_intron_start", matcher.group(6),
1);
        printElement("bdgp_donor_intron", matcher.group(7), 1);
        printElement("/splicesite",0);
    } //end of while loop
} //end of first big if loop

pattern = Pattern.compile(BDGPAcceptorPattern1);
matcher = pattern.matcher(dataXML);
if (matcher.find()) { // results are complete
    String dataRows = matcher.group(1);
    //System.err.println("group 1 = "+matcher.group(1));
    pattern = Pattern.compile(BDGPAcceptorPattern2);
    matcher = pattern.matcher(dataRows);
    while(matcher.find()) {

```

```

        printElement("splicesite",0);
        out.println("\t<bdgp_id>" + CDATA_START +
convertedQuery + CDATA_END + "</bdgp_id>");
        printElement("bdgp_acceptor_start", matcher.group(1), 1);
        printElement("bdgp_acceptor_end", matcher.group(2), 1);
        printElement("bdgp_acceptor_score", matcher.group(3), 1);
        printElement("bdgp_acceptor_intron", matcher.group(4), 1);
        printElement("bdgp_acceptor_intron_end", matcher.group(5),
1);
        printElement("bdgp_acceptor_exon_start", matcher.group(6),
1);
        printElement("bdgp_acceptor_exon", matcher.group(7), 1);
        printElement("/splicesite",0);
    } //end of while loop
} //end of second if loop

    printElement("/bdgp_source",0);

brXML.close();

} //end of try statement

catch (Exception e) {

    out.println(HEADER_XML);
    System.err.println(e.getClass() + ": " + e.getMessage());
    e.printStackTrace();

} //end of catch

} //end of processRequest

public String constructQuery(Iterator queryIterator) {

    StringBuffer termBuf = new StringBuffer("");
    HashMap groupMap = new HashMap();

    String retID = null;

```

```

//out.println("This is the queryIterator: " + queryIterator);

if (queryIterator.hasNext()) {
    while (queryIterator.hasNext()) {
        // Retrieve the term values that were parsed from the query URL.
        HashMap queryMap = (HashMap) queryIterator.next();
        ttype = (String) queryMap.get(AttribOption.OPTION_PARAM_NAME);
        tvalue = (String) queryMap.get(BDGP_OPTION_MAP_TVALUE);
        field = (String) queryMap.get(BDGP_OPTION_MAP_FIELD);
        hexencoded = (String)
            queryMap.get(BDGP_OPTION_MAP_HEXENCODED);

            //out.println("String retID is: " + retID);

            // Convert term values from hex to ascii if necessary.
            if (hexencoded != null && hexencoded.equals("true")) {
                tvalue = StringUtility.convertHexToASCII(tvalue);

                retID = tvalue;
            }

            else {

                retID = tvalue;
            }

        }
    }

//out.println("This is the retID value: " + retID);

return retID;
}

public String getUsage() {
    return(
        HEADER_XML + "\n" +
        "<" + TAG_WRAPPER_SOURCE + ">\n" +

```

```

        " <help>\n" +
        " <usage>\n" +
        " <![CDATA[ \n" + HELP_TEXT +
        " ]]>\n" +
        " </usage>\n" +
        " </help>\n" +
        "</" + TAG_WRAPPER_SOURCE + ">"
    );
}

public void printElement(String tag, String content, int tabs) {
    String tabStr = "";
    content = content.trim();
    for(int o = 0; o < tabs; o++)
        tabStr += "\t";
    out.println(tabStr + "<" + tag + ">" + CDATA_START + content +
        CDATA_END + "</" + tag + ">");
}

public void printElement(String tag, int tabs) {
    String tabStr = "";
    for(int o = 0; o < tabs; o++)
        tabStr += "\t";
    out.println(tabStr + "<" + tag + ">");
}

public void terminate() {
    // TODO Auto-generated method stub
}

}

/*
 *
 * Created on Jul 25, 2006
 *
 *
 */

```

```

* @author Terry Shen, with Peggy's help in Genome Sciences
*           This wrapper uses cookies!
*
*
* TODO To change the template for this generated type comment go to
* Window - Preferences - Java - Code Style - Code Templates
*
*
*/

```

```
package org.biomediator.wrappers;
```

```
import javax.servlet.http.Cookie;
```

```
import org.biomediator.util.AttribOption;
import org.biomediator.util.StringUtility;
```

```
import java.net.URL;
```

```
import java.lang.Object.*;
import java.lang.Class.*;
```

```
import java.io.*;
import java.net.*;
import java.util.*;
import java.util.regex.Matcher;
import java.util.regex.Pattern;
```

```
public class GVSLinkageWrapper extends AbstractWrapper {
```

```

    public static final String TAG_WRAPPER_SOURCE           =
    "GVS_source";
    public static final String HELP_TEXT                   = "
    Here's some help stuff... ";

    public static final String GVS_HELP_TEXT =
    "    Description:\n" +

```

```

        "           Provides an interface for searching the GVS database.\n" +
        "\n" +
        "       Usage:\n" +
        "       Servlet examples:\n" +
        "       Usage:\n" +
        "       Servlet examples:\n" +
        "
?term=term[tvalue=rs405509,hexencoded=false,field=target]" +
"
example=?term=term[tvalue=rs451061,hexencoded=false,field=target]";

/* rs630612
 * chr1:25569579-25569580
 * rs405509
 * 19, chromStart=50100675, chromEnd=50100676
 **/

public static final String GVS_SOURCE_TAG = "gvs_source";
public static final String GVS_CMD_TERM = "term";
public static final String GVS_OPTION_MAP_FIELD = "field";
public static final String GVS_OPTION_MAP_TVALUE = "tvalue";
public static final String GVS_OPTION_MAP_HEXENCODED =
"hexencoded";

public static final String CDATA_START      = "<![CDATA[ ";
public static final String CDATA_END        = " ]]>";
public static final String ENDLINE          = " ";

public static final String prePatternGVS = "<FORM(.*)</FORM>";
public static final String popIDPattern =
"http://www.ncbi.nlm.nih.gov/projects/SNP/snp_viewTable.cgi??pop_id=(.*
?)\>";
public static final String LDPattern = //"#SNP-1 SNP-2 r2<br><!-- output
pairwise r2 value-->((\d+)\s*(\d+)\s*(-
{0,1} {0,1}+\d+\.\d+)<br>)*?</pre>";
"(rs\d+)\s*(rs\d+)\s*(-+\d+\.\d+)<br>";

public String proxyUrl = null;

```

```

private String retQuery = null;

private String ttype = null;
private String tvalue = null;
private String field = null;
private String hexencoded = null;

public String baseURL = "http://gvs.gs.washington.edu/GVS/";
public String maincookie="";
public String mainpopID = "";
public String popID = "";
public String query = "";

// query index.jsp
public static final String url = "http://localhost:8180/GVS/index.jsp";

public static void main(String[] args) {

    GVSLinkageWrapper gvs = new GVSLinkageWrapper();
        //invoking the constructor
    gvs.out = new PrintWriter(System.out, true);
    if (args.length > 0) {
        gvs.processRequest(args[0]);
    }
    else {
        gvs.processRequest("");
    }
}

public void processRequest(String query) {
    //try {
        initializeServlet(HELP_TEXT, TAG_WRAPPER_SOURCE,
getWrapperProperties("gvs"));
        String proxy =
wrapperProperties.getProperty(WrapperProperties.PROP_PROXY_URL);

```

```

AttribOption ao = new AttribOption(query);
processCommands(ao, getWrapperProperties("ucsctissueexp"));

if (proxy == null) {
    proxyUrl = "";
}
else {
    proxyUrl = proxy + "?";
}

//String convertedQuery = null;
TreeSet termTypeSet = ao.getAttribMapSet(GVS_CMD_TERM);

if (termTypeSet != null) {
    retQuery = constructQuery(termTypeSet.iterator());
    //System.out.println ("This is the returned query: " +
retQuery);
    //}

    getURL1(retQuery);
    getURL2(retQuery);
    getURL3(retQuery);
    //getURL4(retQuery);

} //end of if statement

} //end of processRequest method

public void getURL1(String query) {
    try {
        // ---- query index.jsp
        URL mainurl = new URL(baseUrl + "index.jsp");

        URLConnection urlConn1 = mainurl.openConnection();
        urlConn1.connect();

        String headerName=null;

```

```

        for (int i=1; (headerName =
urlConn1.getHeaderFieldKey(i))!=null; i++) {
            if (headerName.equals("Set-Cookie")) {
                maincookie = urlConn1.getHeaderField(i);
                maincookie = maincookie.substring(0,
maincookie.indexOf(";"));
            }
        } //end of for loop
        //System.out.println("Cookie after you connect to the first
page:\n " + maincookie);
    } //end of try
    catch (IOException ioe) {
        ioe.printStackTrace();
    }
} //end of getURL1 method

public void getURL2(String query) {
    // ---- query populationStats.jsp
    try {
        URL rsurl = new URL(baseURL +
"populationStats.jsp?searchType=rsID&upstreamSize=0&downstreamSize=0
");
        //URL chrurl = new URL(baseURL +
"populationStats.jsp?searchType=chromosome&chromosome=1&upstreamSi
ze=0&downstreamSize=0");
        URLConnection urlConn2 = rsurl.openConnection();
        urlConn2.setRequestProperty("Cookie", maincookie);
        urlConn2.connect();

    } //end of try
    catch (IOException ioe) {
        ioe.printStackTrace();
    }
} //end of getURL2 method

public void getURL3(String query) {
    // ---- query populationStats.jsp

```

```

try {
    //retQuery = constructQuery(termTypeSet.iterator());
    //System.out.println("THIS IS THE retQuery:" + retQuery);

    URL popurl = new URL(baseUrl +
"PopStatsServlet?searchBy=dbsnp+rsID&upstreamSize=10000&downstream
Size=10000&x=&y=&" + retQuery);
    //URL popurl = new URL(baseUrl +
"PopStatsServlet?searchBy=chromosome&chromosome=19&chromoStart=5
0100675&chromoEnd=50100676&x=&y=");
    //System.out.println("THIS IS THE URL:" + popurl);

    URLConnection urlConn3 = popurl.openConnection();
    urlConn3.setRequestProperty("Cookie", maincookie);
    urlConn3.connect();

    //System.out.println("Cookie after you connect to the third
page:\n " + maincookie);
    //System.out.println("This is what the URL connection is:\n "
+ urlConn3);

    BufferedReader br = new BufferedReader(new
InputStreamReader(urlConn3.getInputStream()));
    StringBuffer lineBuf = new StringBuffer();
    String inputLine;

    while ((inputLine = br.readLine()) != null) {
        lineBuf.append(inputLine + "\n");
        //System.out.println(inputLine);
    }

    Pattern pattern = Pattern.compile(popIDPattern);
    Matcher matcher = pattern.matcher(lineBuf);

    out.println(HEADER_XML);
    printElement("linkage_disequilibrium",0);
    out.println("\t<source_id>" + CDATA_START + tvalue +
CDATA_END + "</source_id>");

```

```

while (matcher.find()) {
    String pop = matcher.group(1);
    //out.println("This is the pop: " + pop);
    mainpopID = matcher.group(1);
    //popID = ("popID=" + matcher.group(1));
    if (pop.equals("1409")) {
        popID = ("popID=" + pop + "&unrelated=" +
pop);
                                //popID = ("popID=" + matcher.group(1) +
"&unrelated=" + matcher.group(1));
        }
        else if (pop.equals("1412")) {
            popID = ("popID=" + pop + "&unrelated=" +
pop);
                                //popID = ("popID=" + matcher.group(1) +
"&unrelated=" + matcher.group(1));
        }
        else {
            popID = ("popID=" + pop);
            //popID = ("popID=" + matcher.group(1));
        }
        }

        query = popID;

        getURL4(query);

    }

    printElement("/linkage_disequilibrium",0);

} //end of try
catch (IOException ioe) {
    ioe.printStackTrace();
}
} //end of getURL3 method

public void getURL4(String query) {
    // ---- query populationStats.jsp
    try {

```

```

//old url - not working anymore = URL ldurl = new
URL(baseUrl +
"ServletManager?VGWindowMain=first&genotypeSource=dbsnp+rsID&me
rgeSampleAndSnp=1&outputBy=RS_ID&displayBy=Text%2FImage&freq=
5&r2Threshold=0.80&dataCoverageTagSnp=85&dataCoverageClustering=7
0&ldMin=0.1&ldMax=1.0&ldCal.x=52&ldCal.y=12&ldCal=Display+Linka
ge+Disequilibrium&" + query);
URL ldurl = new URL(baseUrl +
"ServletManager?VGWindowMain=first&genotypeSource=dbsnp+rsID&me
rgeSampleAndSnp=1&outputBy=RS_ID&displayBy=Text&freq=5&r2Thres
hold=0.80&dataCoverageTagSnp=85&dataCoverageClustering=70&ldMin=
0.1&ldMax=1.0&ldCal.x=52&ldCal.y=12&ldCal=Display+Linkage+Disequi
librium&" + query);

URLConnection urlConn4 = ldurl.openConnection();
urlConn4.setRequestProperty("Cookie", maincookie);
urlConn4.connect();

BufferedReader br = new BufferedReader(new
InputStreamReader(urlConn4.getInputStream()));
StringBuffer lineBuf = new StringBuffer();
String inputLine;

while ((inputLine = br.readLine()) != null) {
    lineBuf.append(inputLine + "\n");
    //System.out.println(inputLine);
}

//out.println("\t<source_id>" + CDATA_START + tvalue +
CDATA_END + "</source_id>");

Pattern pattern = Pattern.compile(LDPattern);
Matcher matcher = pattern.matcher(lineBuf);

while (matcher.find()) {

    String checksnp1 = (matcher.group(1));
    String checksnp2 = (matcher.group(2));

```

```

String checkr2 = (matcher.group(3));
//out.println("CHECKING VARIABLES!!");
//out.println("checksnp1= " + checksnp1);
//out.println("checksnp2= " + checksnp2);
//out.println("tvalue= " + tvalue);
//out.println("checkr2= " + checkr2);
if(checkr2.equals("0.00")) {
    //out.println(HEADER_XML);
    //out.println("");
}

else if (((checksnp1.equals(tvalue)) ||
(checksnp2.equals(tvalue)))){

    out.println("\t<population_linkage_disequilibrium>");
    out.println("\t<popId>" + CDATA_START +
mainpopID + CDATA_END + "</popId>");
    //out.println("THIS PART WORKS!");
    out.println("\t\t<population_r2>");
    printElement("SNP-1", matcher.group(1),2);
    printElement("SNP-2", matcher.group(2),2);
    printElement("r2", matcher.group(3),2);
    out.println("\t\t</population_r2>");

    out.println("\t</population_linkage_disequilibrium>");

    } //end of else

    } //end of while

    } //end of try
    catch (IOException ioe) {
        //ioe.printStackTrace();
    }

} //end of getURL3 method

```

```

public String constructQuery(Iterator queryIterator) {

    StringBuffer termBuf = new StringBuffer("");
    HashMap groupMap = new HashMap();

    //Example of what the parameter result should be:
    //term=term[tvalue=chr1:2107228-
    2107230,hexencoded=false,field=position] ;

    while (queryIterator.hasNext()) {
        // Retrieve the term values that were parsed from the query
        URL.
        HashMap queryMap = (HashMap) queryIterator.next();
        ttype = (String)
        queryMap.get(AttribOption.OPTION_PARAM_NAME);
        tvalue = (String)
        queryMap.get(GVS_OPTION_MAP_TVALUE);
        field = (String) queryMap.get(GVS_OPTION_MAP_FIELD);
        hexencoded = (String)
        queryMap.get(GVS_OPTION_MAP_HEXENCODED);

        // Convert term values from hex to ascii if necessary.
        if (hexencoded != null && hexencoded.equals("true")) {
            tvalue = StringUtility.convertHexToASCII(tvalue);
            //dbvalue =
            StringUtility.convertHexToASCII(dbvalue);
        }

        retQuery = field + "=" + tvalue;
    }

    //System.out.println ("This is the returned query: " + retQuery);
    return retQuery;
} //end of constructQuery Method

/**
 * getUsage
 * @return
 */

```

```

public String getUsage() {
    return(
        HEADER_XML + "\n" +
        "<" + TAG_WRAPPER_SOURCE + ">\n" +
        " <help>\n" +
        " <usage>\n" +
        " <![CDATA[ \n" + HELP_TEXT +
        " ]>\n" +
        " </usage>\n" +
        " </help>\n" +
        "</" + TAG_WRAPPER_SOURCE + ">"
    );
}

public void printElement(String tag, String content, int tabs) {
    String tabStr = "";
    content = content.trim();
    for(int o = 0; o < tabs; o++)
        tabStr += "\t";
    out.println(tabStr + "<" + tag + ">" + CDATA_START + content +
        CDATA_END + "</" + tag + ">");
}

public void printElement(String tag, int tabs) {
    String tabStr = "";
    for(int o = 0; o < tabs; o++)
        tabStr += "\t";
    out.println(tabStr + "<" + tag + ">");
}

public void terminate() {
    // TODO Auto-generated method stub
}

} //end of GVSLinkage class
/*
 * Created on Jul 14, 2006
 *
 **

```

```

* @author Terry Shen, again with borrowing PFam code and John's help :)
*
* This wrapper uses HashMap to capture the query string.
*
*
* TODO To change the template for this generated type comment go to
* Window - Preferences - Java - Code Style - Code Templates
*
*
*/

package org.biomediator.wrappers;

import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.io.PrintWriter;
import java.net.URL;
import java.net.URLDecoder;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.Iterator;
import java.util.StringTokenizer;
import java.util.TreeSet;
import java.util.regex.Matcher;
import java.util.regex.Pattern;

import org.biomediator.util.AttribOption;
import org.biomediator.util.StringUtility;

import com.oroinc.text.regex.Perl5StreamInput;

public class HaplotterWrapper extends AbstractWrapper {

    public static final String HAPLOTTER_HELP_TEXT
        = "      Description:\n" +
        "      Provides an interface for searching the Haplotter database.\n" +
        "\n" +
        "      Usage:\n" +
        "      Servlet examples:\n" +

```

```
"
    ?term=term[tvalue=rs405509,hexencoded=false,field=target,sz=sz,szvalue=5]";
```

```
public static final String HAPLOTTER_SOURCE_TAG =
"HAPLOTTER_source";
public static final String HAPLOTTER_CMD_TERM = "term";
public static final String HAPLOTTER_OPTION_MAP_FIELD = "field";
public static final String HAPLOTTER_OPTION_MAP_TVALUE =
"tvalue";
public static final String HAPLOTTER_OPTION_MAP_DB = "db";
public static final String HAPLOTTER_OPTION_MAP_SZ = "sz";
public static final String HAPLOTTER_OPTION_MAP_DBVALUE =
"dbvalue";
public static final String HAPLOTTER_OPTION_MAP_SZVALUE =
"szvalue";
public static final String HAPLOTTER_OPTION_MAP_HEXENCODED =
"hexencoded";
```

```
public static final String CDATA_START      = "<![CDATA[ ";
public static final String CDATA_END        = " ]]>";
public static final String ENDLINE          = " ";
public static final String prePattern       = "http://hg-
wen.uchicago.edu/cgi-bin/ihh.cgi.(*?)</html>";
```

```
public static final String urlBase = "http://hg-wen.uchicago.edu/cgi-
bin/ihh.cgi?";
public static final String urlQuery = "chr=&action=1&ht=0&";
```

```
public static final String sourceID = "NAME=target VALUE=(.*)><INPUT
TYPE=HIDDEN NAME=db";
public static final String ethnicGroupDB = "NAME=db
VALUE=(.*)><INPUT TYPE=HIDDEN NAME=src1";
public static final String derivedAlleleFrequency = "NAME=cfreq
VALUE=(\\d+\\.\\d*)"; //regular expression for decimal numbers
public static final String standardizediHS = "NAME=ihh
VALUE=(\\d+\\.\\d*)";
public static final String fayWuH = "NAME=H VALUE=(^+|-?\\d+\\.\\d*)";
public static final String fst1 = "NAME=fst1 VALUE=([a-z]+:[a-
z]+:\\d+\\.\\d*)";
```

```
public static final String fst2 = "NAME=fst2 VALUE=([a-z]+:[a-z]+:\\d+\\.\\d*)";
public static final String chr = "NAME=chr VALUE=([0-9]*)";
```

```
public String proxyUrl = null;
private Pattern pattern = null;
private Matcher matcher = null;
private HashMap queryMap = null;
```

```
private String ttype = null;
private String tvalue = null;
private String field = null;
private String hexencoded = null;
private String retQuery = null;
private String db = null;
private String dbvalue = null;
private String sz = null;
private String szvalue = null;
```

```
/**
 * main
 * @param args
 */
public static void main(String[] args) {
    HaplotterWrapper w = new HaplotterWrapper();
    //invoking the constructor
    w.out = new PrintWriter(System.out, true);
    if (args.length > 0) {
        w.processRequest(args[0]);
    }
    else {
        w.processRequest("");
    }
} //end of main method
```

```
/**
 * processRequest
 * @param query
```

```

*/

public void processRequest(String query) {
    initializeServlet(HAPLOTTER_HELP_TEXT,
        HAPLOTTER_SOURCE_TAG, getWrapperProperties("haplotter"));

    // Parse the query string and process commands.
    if (query == null || query.equals("")) {
        query = CMD_HELP;
    }

    // Check to see if we should exit immediately.
    if (isExitCmd) {
        return;
    }

    String proxy =
wrapperProperties.getProperty(WrapperProperties.PROP_PROXY_URL);

    doQuery(query);

} //end of ProcessRequest method

public void doQuery(String query) {

    AttribOption ao = new AttribOption(query);
    processCommands(ao, getWrapperProperties("haplotter"));

    String proxy =
wrapperProperties.getProperty(WrapperProperties.PROP_PROXY_URL);
    if (proxy == null) {
        proxyUrl = "";
    }
    else {
        proxyUrl = proxy + "?";
    }

    TreeSet termTypeSet =
ao.getAttribMapSet(HAPLOTTER_CMD_TERM);

```

```

if (termTypeSet != null) {
    retQuery = constructQuery(termTypeSet.iterator());
}

try {

    // Download HTML page for each of the ethnic groups.
    //String urlceu = urlBase + urlQuery + field + "=" + tvalue +
    "&" + sz + "=" + szvalue + "&" + "db" + "=" + "ceu";
    //String urlryi = urlBase + urlQuery + field + "=" + tvalue +
    "&" + sz + "=" + szvalue + "&" + "db" + "=" + "ryi";
    //String urlasn = urlBase + urlQuery + field + "=" + tvalue +
    "&" + sz + "=" + szvalue + "&" + "db" + "=" + "asn";
    String urlceu = proxyUrl + urlBase + urlQuery + field + "=" +
    tvalue + "&" + sz + "=" + szvalue + "&" + "db" + "=" + "ceu";
    String urlryi = proxyUrl + urlBase + urlQuery + field + "=" +
    tvalue + "&" + sz + "=" + szvalue + "&" + "db" + "=" + "ryi";
    String urlasn = proxyUrl + urlBase + urlQuery + field + "=" +
    tvalue + "&" + sz + "=" + szvalue + "&" + "db" + "=" + "asn";

    // Download HTML page.
    BufferedReader brceu = new BufferedReader(new
    InputStreamReader(new URL(urlceu).openStream()));
    //BufferedReader bryri = new BufferedReader(new
    InputStreamReader(new URL(urlryi).openStream()));
    //BufferedReader brasn = new BufferedReader(new
    InputStreamReader(new URL(urlasn).openStream()));
    StringBuffer lineBufceu = new StringBuffer();
    //StringBuffer lineBufryi = new StringBuffer();
    //StringBuffer lineBufasn = new StringBuffer();
    String lineceu = null;
    //String lineyri = null;
    //String lineasn = null;

    out.println(HEADER_XML);
    printElement("haplotter_total",0);

    out.println("<source_id>" + CDATA_START + tvalue +
    CDATA_END + "</source_id>");

```

```

while ((lineceu = brceu.readLine()) != null) {
    lineBufceu.append(lineceu);
}

//out.println("This is the line buffer being taken in: " +
lineBufceu);

pattern = Pattern.compile(prePattern);
matcher = pattern.matcher(lineBufceu);

if(matcher.find()) { // results are complete

    //printElement("haplotter",1);

    pattern = Pattern.compile(ethnicGroupDB);
    matcher = pattern.matcher(lineBufceu);

    while(matcher.find()) {
        printElement("haplotter",1);
        printElement("ethnic_group",
matcher.group(1),2);
    }
    pattern = Pattern.compile(derivedAlleleFrequency);
    matcher = pattern.matcher(lineBufceu);
    while(matcher.find()) {
        printElement("derived_allele_frequency",
matcher.group(1),2);
    }
    pattern = Pattern.compile(standardizediHS);
    matcher = pattern.matcher(lineBufceu);
    while(matcher.find()) {
        printElement("standardized_iHS",
matcher.group(1),2);
    }
    pattern = Pattern.compile(fayWuH);
    matcher = pattern.matcher(lineBufceu);
    while(matcher.find()) {
        printElement("fay_wu_H",
matcher.group(1),2);
    }
}

```

```

pattern = Pattern.compile(fst1);
matcher = pattern.matcher(lineBufceu);
while(matcher.find()) {
    printElement("fst1", matcher.group(1),2);
}
pattern = Pattern.compile(fst2);
matcher = pattern.matcher(lineBufceu);
while(matcher.find()) {
    printElement("fst2", matcher.group(1),2);
}
pattern = Pattern.compile(chr);
matcher = pattern.matcher(lineBufceu);
while(matcher.find()) {
    printElement("chr", matcher.group(1),2);
    printElement("/haplotter",1);
}

//printElement("/haplotter",1);

//brceu.close();

} //end of all ceu if statements

```

```

BufferedReader bryri = new BufferedReader(new
InputStreamReader(new URL(urlyri).openStream()));
StringBuffer lineBufyri = new StringBuffer();
String lineyri = null;

while ((lineyri = bryri.readLine()) != null) {
    lineBufyri.append(lineyri);
}

pattern = Pattern.compile(prePattern);
matcher = pattern.matcher(lineBufyri);

if(matcher.find()) { // results are complete

    //printElement("haplotter",1);

```

```

pattern = Pattern.compile(ethnicGroupDB);
matcher = pattern.matcher(lineBufyri);

while(matcher.find()) {
    printElement("haplotter",1);
    printElement("ethnic_group",
matcher.group(1),2);
}
pattern = Pattern.compile(derivedAlleleFrequency);
matcher = pattern.matcher(lineBufyri);
while(matcher.find()) {
    printElement("derived_allele_frequency",
matcher.group(1),2);
}
pattern = Pattern.compile(standardizediHS);
matcher = pattern.matcher(lineBufyri);
while(matcher.find()) {
    printElement("standardized_iHS",
matcher.group(1),2);
}
pattern = Pattern.compile(fayWuH);
matcher = pattern.matcher(lineBufyri);
while(matcher.find()) {
    printElement("fay_wu_H",
matcher.group(1),2);
}
pattern = Pattern.compile(fst1);
matcher = pattern.matcher(lineBufyri);
while(matcher.find()) {
    printElement("fst1", matcher.group(1),2);
}
pattern = Pattern.compile(fst2);
matcher = pattern.matcher(lineBufyri);
while(matcher.find()) {
    printElement("fst2", matcher.group(1),2);
}
pattern = Pattern.compile(chr);
matcher = pattern.matcher(lineBufyri);
while(matcher.find()) {
    printElement("chr", matcher.group(1),2);
    printElement("/haplotter",1);
}

```

```

    }

    //printElement("/haplotter",1);
    //bryri.close();

} //end of all yri if statements

BufferedReader brasn = new BufferedReader(new
InputStreamReader(new URL(ulasn).openStream()));
StringBuffer lineBufasn = new StringBuffer();
String lineasn = null;

while ((lineasn = brasn.readLine()) != null) {
    lineBufasn.append(lineasn);
}

pattern = Pattern.compile(prePattern);
matcher = pattern.matcher(lineBufasn);

if(matcher.find()) { // results are complete

    //printElement("haplotter",1);

    pattern = Pattern.compile(ethnicGroupDB);
    matcher = pattern.matcher(lineBufasn);
    while(matcher.find()) {
        printElement("haplotter",1);
        printElement("ethnic_group",
matcher.group(1),2);
    }
    pattern = Pattern.compile(derivedAlleleFrequency);
    matcher = pattern.matcher(lineBufasn);
    while(matcher.find()) {
        printElement("derived_allele_frequency",
matcher.group(1),2);
    }
    pattern = Pattern.compile(standardizediHS);
    matcher = pattern.matcher(lineBufasn);
    while(matcher.find()) {

```

```

        printElement("standardized_iHS",
matcher.group(1),2);
    }
    pattern = Pattern.compile(fayWuH);
    matcher = pattern.matcher(lineBufasn);
    while(matcher.find()) {
        printElement("fay_wu_H",
matcher.group(1),2);
    }
    pattern = Pattern.compile(fst1);
    matcher = pattern.matcher(lineBufasn);
    while(matcher.find()) {
        printElement("fst1", matcher.group(1),2);
    }
    pattern = Pattern.compile(fst2);
    matcher = pattern.matcher(lineBufasn);
    while(matcher.find()) {
        printElement("fst2", matcher.group(1),2);
    }
    pattern = Pattern.compile(chr);
    matcher = pattern.matcher(lineBufasn);
    while(matcher.find()) {
        printElement("chr", matcher.group(1),2);
        printElement("/haplotter",1);
    }

    //printElement("/haplotter",1);
    //brasn.close();

} //end of all asn if statements

//printElement("/source_id",0);
printElement("/haplotter_total",0);

} //end of try

catch (Exception e) {

```

```

        out.println(HEADER_XML);
        System.err.println(e);
    } //end of catch
} //end of doQuery Method

public String constructQuery(Iterator queryIterator) {

    StringBuffer termBuf = new StringBuffer("");
    HashMap groupMap = new HashMap();

    //Example of what the parameter result should be:

    //term=term[tvalue=rs405509,hexencoded=false,field=target,db=db,d
    bvalue=ceu,sz=sz,szvalue=5]" ;

    while (queryIterator.hasNext()) {
        // Retrieve the term values that were parsed from the query
        URL.
        HashMap queryMap = (HashMap) queryIterator.next();
        ttype = (String)
        queryMap.get(AttribOption.OPTION_PARAM_NAME);
        tvalue = (String)
        queryMap.get(HAPLOTTER_OPTION_MAP_TVALUE);
        field = (String)
        queryMap.get(HAPLOTTER_OPTION_MAP_FIELD);
        db = (String)
        queryMap.get(HAPLOTTER_OPTION_MAP_DB);
        dbvalue = (String)
        queryMap.get(HAPLOTTER_OPTION_MAP_DBVALUE);
        sz = (String)
        queryMap.get(HAPLOTTER_OPTION_MAP_SZ);
        szvalue = (String)
        queryMap.get(HAPLOTTER_OPTION_MAP_SZVALUE);

        hexencoded = (String)
        queryMap.get(HAPLOTTER_OPTION_MAP_HEXENCODED);
    }
}

```

```

        // Convert term values from hex to ascii if necessary.
        if (hexencoded != null && hexencoded.equals("true")) {
            tvalue = StringUtility.convertHexToASCII(tvalue);
        }

        //retQuery = field + "=rs" + tvalue + "&" + db + "=" +
        dbvalue+ "&" + sz + "=" + szvalue;    //if you wanted to take out the rs
        //retQuery = field + "=" + tvalue + "&" + db + "=" + dbvalue+
        "&" + sz + "=" + szvalue;
    }

    //System.out.println (retQuery);
    return tvalue;
    //return retQuery;
} //end of constructQuery Method

```

```

public HashMap buildQueryMap (String query) {
    System.out.println(" decoded: "+ query);
    HashMap queryMap = new HashMap();
    StringTokenizer st = new StringTokenizer(query, "&");
    while (st.hasMoreTokens())
    {
        StringTokenizer st1 = new StringTokenizer(st.nextToken(),
"=");
        String key = st1.nextToken();
        String value = (st1.hasMoreTokens()?st1.nextToken():"");
        //System.out.println(" key: "+key+", value: "+value);
        queryMap.put(key, new String[]{value});
    }
    return queryMap;
}

```

```

public void printElement(String tag, String content, int tabs) {
    String tabStr = "";
    content = content.trim();
    for(int o = 0; o < tabs; o++)
        tabStr += "\t";
}

```

```

        out.println(tabStr + "<" + tag + ">" + CDATA_START + content +
CDATA_END + "</" + tag + ">");
    }

    public void printElement(String tag, int tabs) {
        String tabStr = "";
        for(int o = 0; o < tabs; o++)
            tabStr += "\t";
        out.println(tabStr + "<" + tag + ">");
    }

    /**
     * getUsage
     * @return
     */
    public String getUsage() {
        return(
            HEADER_XML + "\n" +
            "<" + TAG_WRAPPER_SOURCE + ">\n" +
            " <help>\n" +
            " <usage>\n" +
            " <![CDATA[ \n" + HELP_TEXT +
            " ]]>\n" +
            " </usage>\n" +
            " </help>\n" +
            "</" + TAG_WRAPPER_SOURCE + ">"
        );
    }

    public void terminate() {
        // TODO Auto-generated method stub
    }

}

/*
 * Created on Jul 10, 2006
 */

```

```

* @author Terry Shen with lots of "borrowing" of other code and aid from John
*
* Doesn't use HashMap
*
*
* TODO To change the template for this generated type comment go to
* Window - Preferences - Java - Code Style - Code Templates
*/
package org.biomediator.wrappers;

import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.io.PrintWriter;
import java.net.URL;
import java.util.HashMap;
import java.util.Iterator;
import java.util.TreeSet;
import java.util.regex.Matcher;
import java.util.regex.Pattern;

import org.biomediator.util.AttribOption;
import org.biomediator.util.StringUtility;

import com.oroinc.text.regex.Perl5StreamInput;

public class HGMDGeneWrapper extends AbstractWrapper {

    public static final String HGMD_HELP_TEXT =
        "      Description:\n" +
        "      Provides an interface for searching the HGMD database.\n" +
        "\n" +
        "      Usage:\n" +
        "      Servlet examples:\n" +
        "
        ?term=term[tvalue=APOE,hexencoded=false,field=gene]" ;

    public static final String HGMD_SOURCE_TAG = "hgmd_source";
    public static final String HGMD_CMD_TERM = "term";
    public static final String HGMD_OPTION_MAP_FIELD = "field";

```

```

public static final String HGMD_OPTION_MAP_TVALUE = "tvalue";
public static final String HGMD_OPTION_MAP_HEXENCODED =
    "hexencoded";
//public static final int MAX_DOCSIZE = 1024 * 10000;

public static final String CDATA_START      = "<![CDATA[ ";
public static final String CDATA_END        = " ]]>";
public static final String ENDLINE          = " ";
public static final String prePattern      = "<td><b>Gene
    name.*?</html>";

//public static final String gene = "<td align=center>" + GENE + "</td><td
    align=center>([^\<]*?)</td><td align=left>([^\<]*?)</td>";
//<td
    align=center>APOE</td><td align=center>19q13.2</td><td
    align=left>Apolipoprotein E</td>
public static final String gene = "<td align=center>([^\<]*?)</td><td
    align=center>([^\<]*?)</td><td align=left>([^\<]*?)</td>";
public static final String missense = "<td>Missense/nonsense</td><td
    align=center>([0-9]*)</td>";
public static final String splicing = "<td>Splicing </td><td align=center>([0-
    9]*)</td>";
public static final String regulatory = "<td>Regulatory </td><td align=center>([0-
    9]*)</td>";
public static final String smallDeletions = "<td>Small deletions </td><td
    align=center>([0-9]*)</td>";
public static final String smallInsertions = "<td>Small insertions </td><td
    align=center>([0-9]*)</td>";
public static final String smallIndels = "<td>Small indels </td><td
    align=center>([0-9]*)</td>";
public static final String grossDeletions = "<td>Gross deletions </td><td
    align=center>([0-9]*)</td>";
public static final String grossInsertions = "<td>Gross insertions </td><td
    align=center>([0-9]*)</td>";
public static final String complexRearrangements = "<td>Complex
    rearrangements </td><td align=center>([0-9]*)</td>";
public static final String repeatVariations = "<td>Repeat variations </td><td
    align=center>([0-9]*)</td>";
public static final String total = "<b>([0-9]*)</b> \\\([0-9]*\\)</td>";
//public static final String diseasePhenotype = "Mutation data by
    disease/phenotype.*?</table>";

```

```

public static final String diseasePhenotype = "<td>([^\<>]*)?</td><td
    align=center>[0-9]*</td><td align=center><a href='http://www.biobase-
    international.com/'>";

/*
 * [^\<>]*? = regular expression that means Reluctant quantifier of any character
    that's not the "<>" signs
 * [0-9]* = regular expression that means any number 0-9 repeated once or more
 */

public String proxyUrl = null;
private Pattern pattern = null;
private Matcher matcher = null;
private HashMap queryMap = null;

public static void main(String[] args) {

    HGMDGeneWrapper hgmd = new HGMDGeneWrapper();
        //invoking the constructor
    hgmd.out = new PrintWriter(System.out, true);
    if (args.length > 0) {
        hgmd.processRequest(args[0]);
    }
    else {
        hgmd.processRequest("");
    }
}

public void processRequest(String query) {

    // Make sure the servlet is initialized.
    initializeServlet(HGMD_HELP_TEXT, HGMD_SOURCE_TAG,
        getWrapperProperties("hgmd"));

    // Parse the query string and process commands.
    if (query == null || query.equals("")) {
        query = CMD_HELP;
    }
}

```

```

AttribOption ao = new AttribOption(query);
processCommands(ao, getWrapperProperties("hgmd"));

// Check to see if we should exit immediately.
if (isExitCmd) {
    return;
}

String proxy =
    wrapperProperties.getProperty(WrapperProperties.PROP_PROXY_URL);
if (proxy == null) {
    proxyUrl = "";
}
else {
    proxyUrl = proxy + "?";
}

String convertedQuery = null;
TreeSet termTypeSet = ao.getAttribMapSet(HGMD_CMD_TERM);
if (termTypeSet != null) {
    convertedQuery = constructHGMDQuery(termTypeSet.iterator());
}

String urlBase = "http://www.hgmd.cf.ac.uk/ac/gene.php?";

try {

    // Download HTML page.
    String queryURL = proxyUrl + urlBase + convertedQuery;

    //System.out.println(queryURL);
    BufferedReader br = new BufferedReader(new InputStreamReader(new
    URL(queryURL).openStream()));
    StringBuffer lineBuf = new StringBuffer();
    String line = null;

    while ((line = br.readLine()) != null) {
        lineBuf.append(line);
    }
}

```

```
pattern = Pattern.compile(prePattern);
matcher = pattern.matcher(lineBuf);

//System.out.println(lineBuf);

if(matcher.find()) { // results are complete

    out.println(HEADER_XML);

    //System.out.println("This part works");
    //String preBlock = matcher.group(1);

    printElement("hgmd",0);

    pattern = Pattern.compile(gene);
    matcher = pattern.matcher(lineBuf);
    if(matcher.find()) {
        printElement("gene_symbol", matcher.group(1),1);
        printElement("gene_location", matcher.group(2),1);
        printElement("gene_name", matcher.group(3),1);
    }
    pattern = Pattern.compile(missense);
    matcher = pattern.matcher(lineBuf);
    if(matcher.find()) {
        printElement("missense_nonsense", matcher.group(1),1);
    }
    pattern = Pattern.compile(splicing);
    matcher = pattern.matcher(lineBuf);
    if(matcher.find()) {
        printElement("splicing", matcher.group(1),1);
    }
    pattern = Pattern.compile(regulatory);
    matcher = pattern.matcher(lineBuf);
    if(matcher.find()) {
        printElement("regulatory", matcher.group(1),1);
    }
    pattern = Pattern.compile(smallDeletions);
    matcher = pattern.matcher(lineBuf);
    if(matcher.find()) {
        printElement("small_deletions", matcher.group(1),1);
    }
}
```

```
pattern = Pattern.compile(smallInsertions);
matcher = pattern.matcher(lineBuf);
if(matcher.find()) {
    printElement("small_insertions", matcher.group(1),1);
}
pattern = Pattern.compile(smallIndels);
matcher = pattern.matcher(lineBuf);
if(matcher.find()) {
    printElement("small_indels", matcher.group(1),1);
}
pattern = Pattern.compile(grossDeletions);
matcher = pattern.matcher(lineBuf);
if(matcher.find()) {
    printElement("gross_deletions", matcher.group(1),1);
}
pattern = Pattern.compile(grossInsertions);
matcher = pattern.matcher(lineBuf);
if(matcher.find()) {
    printElement("gross_insertions", matcher.group(1),1);
}
pattern = Pattern.compile(complexRearrangements);
matcher = pattern.matcher(lineBuf);
if(matcher.find()) {
    printElement("complex_rearrangements", matcher.group(1),1);
}
pattern = Pattern.compile(repeatVariations);
matcher = pattern.matcher(lineBuf);
if(matcher.find()) {
    printElement("repeat_variations", matcher.group(1),1);
}
pattern = Pattern.compile(total);
matcher = pattern.matcher(lineBuf);
if(matcher.find()) {
    printElement("total_mutations", matcher.group(1),1);
}
pattern = Pattern.compile(diseasePhenotype);
matcher = pattern.matcher(lineBuf);
while(matcher.find()) {
    printElement("disease_phenotype", matcher.group(1),1);
}
```

```

        printElement("/hgmd",0);

    } //end of all if statements

    br.close();
    out.println(HEADER_XML);
    out.println("No record found in HGMD");

} //end of try

catch (Exception e) {
    // Process remainder of query.
    out.println(HEADER_XML);
    out.println("No record found in HGMD");
    System.err.println(e);
} //end of catch

} //end of processRequest

public void printElement(String tag, String content, int tabs) {
    String tabStr = "";
    content = content.trim();
    for(int o = 0; o < tabs; o++)
        tabStr += "\t";
    out.println(tabStr + "<" + tag + ">" + CDATA_START + content +
        CDATA_END + "</" + tag + ">");
    //System.out.println(tabStr + "<" + tag + ">" + CDATA_START + content +
        CDATA_END + "</" + tag + ">");
}

public void printElement(String tag, int tabs) {
    String tabStr = "";
    for(int o = 0; o < tabs; o++)
        tabStr += "\t";
    out.println(tabStr + "<" + tag + ">");
    //System.out.println(tabStr + "<" + tag + ">");
}

/**
 * getUsage
 * @return

```

```

*/
public String getUsage() {
    return(
        HEADER_XML + "\n" +
        "<" + TAG_WRAPPER_SOURCE + ">\n" +
        " <help>\n" +
        " <usage>\n" +
        " <![CDATA[ \n" + HELP_TEXT +
        " ]]>\n" +
        " </usage>\n" +
        " </help>\n" +
        "</" + TAG_WRAPPER_SOURCE + ">"
    );
}

public String constructHGMDQuery(Iterator queryIterator) {

    StringBuffer termBuf = new StringBuffer("");
    HashMap groupMap = new HashMap();
    String ttype = null;
    String tvalue = null;
    String field = null;
    String hexencoded = null;
    String retQuery = null;

    //term=term[tvalue=APOE,hexencoded=false,field=gene]
    while (queryIterator.hasNext()) {
        // Retrieve the term values that were parsed from the query URL.
        HashMap queryMap = (HashMap) queryIterator.next();
        ttype = (String) queryMap.get(AttribOption.OPTION_PARAM_NAME);
        tvalue = (String) queryMap.get(HGMD_OPTION_MAP_TVALUE);
        field = (String) queryMap.get(HGMD_OPTION_MAP_FIELD);

        hexencoded = (String)
            queryMap.get(HGMD_OPTION_MAP_HEXENCODED);

        // Convert term values from hex to ascii if necessary.
        if (hexencoded != null && hexencoded.equals("true")) {
            tvalue = StringUtility.convertHexToASCII(tvalue);
        }
    }
}

```

```
    }

    retQuery = field + "=" + tvalue;
}

//System.out.println (retQuery);
return retQuery;
}

public void terminate() {
    // TODO Auto-generated method stub

}

}

/**
 * @author Terry Shen, same as all the others :)
 *      This website does POST instead of GET.
 *
 * Created on Jul 19, 2006
 *
 * TODO To change the template for this generated file go to
 * Window - Preferences - Java - Code Style - Code Templates
 */

package org.biomediator.wrappers;

import java.io.*;
import java.net.*;
import java.util.*;

import java.util.regex.Matcher;
import java.util.regex.Pattern;

import org.biomediator.util.AttribOption;
import org.biomediator.util.StringUtility;
```

```

public class SIFTWrapper extends AbstractWrapper {

    public static final String SIFT_HELP_TEXT =
        "    Description:\n" +
        "    Provides an interface for searching the SIFT database.\n" +
        "\n" +
        "    Usage:\n" +
        "    Servlet examples:\n" +
        "        ?term=term[tvalue=rs1098, rs14115,
        hexencoded=false,field=snp]" ;

    public static final String SIFT_SOURCE_TAG = "sift_source";
    public static final String SIFT_CMD_TERM = "term";
    public static final String SIFT_OPTION_MAP_FIELD = "field";
    public static final String SIFT_OPTION_MAP_TVALUE = "tvalue";
    public static final String SIFT_OPTION_MAP_HEXENCODED = "hexencoded";

    public static final String CDATA_START = "<![CDATA[ ";
    public static final String CDATA_END = " ]]>";
    public static final String ENDLINE = " ";
    public static final String prePattern = "<TABLE
        BORDER>(.*?)</TABLE>";

    public static final String urlBase = "http://blocks.fhcrc.org/sift-
        bin/SIFT_dbSNP_submit.pl";

    public static final String mainPattern =
        "<TR><TD ROWSPAN=2>(^[^<>]*)</TD>"+
        "    //SNP, 1
        <TD ROWSPAN=2>(^[^<>]*)</TD>"+
        "    //Amino acid change, 2
        <TD ROWSPAN=2>(^[^<>]*)</TD>"+
        "    //Protein, 3
        <TD>(^[^<>]*)</TD>"+
        "    //amino_acid_1, 4

```

```

"<TD>(<font color=red>|)([^\>]*?)(</font>|)</TD>" +
  //prediction_1_orthologue, 6
"<TD>([^\>]*?)</TD>" +
  //score_1_orthologue, 8
"<TD ROWSPAN=2>([^\>]*?)(.*?)</TD>" + "<TD ROWSPAN=2><A
HREF=([^\>]*?)TARGET=_blank>Alignment of best hits</A>" +
  //Median Sequence IC_orthologue and Alignment_orthologue, 9,11
"</TD><TD>(<font color=red>|)([^\>]*?)(</font>|)</TD>" +
  //prediction_1_homologue, 13
"<TD>([^\>]*?)</TD>" +
  //score_1_homologue, 15
"<TD ROWSPAN=2>([^\>]*?)(.*?)</TD>" + "<TD ROWSPAN=2><A
HREF=([^\>]*?)TARGET=_blank>Alignment of all hits</A>" +
"</TD></TR><TR><TD>([^\>]*?)</TD>" + //Median Sequence
IC_orthologue and Alignment_orthologue and amino_acid_2, 16,18,19

"<TD>(<font color=red>|)([^\>]*?)(</font>|)</TD>" +
  //prediction_2_orthologue,21
"<TD>([^\>]*?)</TD>" +
  //score_2_orthologue,23
"<TD>(<font color=red>|)([^\>]*?)(</font>|)</TD>" +
  //prediction_2_homologue,25
"<TD>([^\>]*?)</TD>";
  //score_2_homologue,27

```

```

/* SAMPLE SOURCE

```

```

<TR><TD ROWSPAN=2>rs1098</TD>
<TD ROWSPAN=2>Q8L</TD>
<TD ROWSPAN=2>NP_078974</TD>
<TD>Q</TD>

<TD><font color=red>DAMAGING</font></TD>
<TD>0.00</TD>
<TD ROWSPAN=2>3.27<BR><i><font color=red>Warning! Low
confidence.*</font></i></TD><TD ROWSPAN=2><A
HREF="http://blocks.fhrc.org/sift-
bin/catfile.csh?/home/sift/tmp/NP_078974.besthits.msf+Alignment+PRE"
TARGET=_blank>Alignment of best hits</A>
</TD><TD><font color=red>DAMAGING</font></TD>

```

```

<TD>0.00</TD>
<TD ROWSPAN=2>3.03</TD><TD ROWSPAN=2><A
HREF="http://blocks.fhrc.org/sift-
bin/catfile.csh?/home/sift/tmp/NP_078974.allhits.msf+Alignment+PRE"
TARGET=_blank>Alignment of all
hits</A></TD></TR><TR><TD>L</TD>

```

```

<TD>TOLERATED</TD>
<TD>1.00</TD>
<TD>TOLERATED</TD>
<TD>1.00</TD>
</TR>

```

```
*/
```

```

public String proxyUrl = null;
private Pattern pattern = null;
private Matcher matcher = null;
private HashMap queryMap = null;

```

```
private String retQuery = null;
```

```
/**
```

```
* main
```

```
* @param args
```

```
*/
```

```

public static void main(String[] args) {
    SIFTWrapper w = new SIFTWrapper();
    constructor
    w.out = new PrintWriter(System.out, true);
    if (args.length > 0) {
        w.processRequest(args[0]);
    }
    else {
        w.processRequest("");
    }
}

```

```
} //end of main method
```

```
//invoking the
```

```
/**
```

```

* processRequest
* @param query
*/
public void processRequest(String query) {
    // Make sure the servlet is initialized.
    initializeServlet(SIFT_HELP_TEXT, SIFT_SOURCE_TAG,
        getWrapperProperties("SIFT"));

    // Parse the query string and process commands.
    if (query == null || query.equals("")) {
        query = CMD_HELP;
    }

    // Check to see if we should exit immediately.
    if (isExitCmd) {
        return;
    }

    //String proxy =
    wrapperProperties.getProperty(WrapperProperties.PROP_PROXY_URL);

    doQuery(query);

} //end of ProcessRequest method

public void doQuery(String query) {

    AttribOption ao = new AttribOption(query);
    processCommands(ao, getWrapperProperties("sift"));

    String proxy =
        wrapperProperties.getProperty(WrapperProperties.PROP_PROXY_URL);
    if (proxy == null) {
        proxyUrl = "";
    }
    else {
        proxyUrl = proxy + "?";
    }

    String convertedQuery = null;

```

```

TreeSet termTypeSet = ao.getAttribMapSet(SIFT_CMD_TERM);
if (termTypeSet != null) {
    convertedQuery = constructQuery(termTypeSet.iterator());
}
}

try {

    // Download HTML page.
    //String url = urlBase + convertedQuery;

    String snp = retQuery; //this line is from the older version
    System.out.println ("This is the returned query after you call it in doQuery: "
+ snp);
    URL destURL = new URL(proxyUrl + urlBase);
    System.out.println("URL destURL: " + destURL);

    URLConnection urlConn = destURL.openConnection();

    //Tell the server what kind of data you are sending - in this case, just a stream
of bytes.
    urlConn.setRequestProperty("Content-Type", "application/x-www-form-
urlencoded");

    //Must tell the server the size of the data you are sending. This also
//tells the URLConnection class that you are doing a POST instead
//of a GET.
    urlConn.setRequestProperty("Content-length", Integer.toString(snp.length()
));

    urlConn.setDoOutput(true);

    //Open an output stream so you can send the info you are posting
    DataOutputStream outputStream = new
    DataOutputStream(urlConn.getOutputStream());

    //Write out the actual request data
    String content = "SNP=" + URLEncoder.encode (snp);
    outputStream.writeBytes(content);
    outputStream.close();
}

```

```
//Now that you have sent the data, open up an input stream and get
//the response back from the server
//DataInputStream inStream = new
DataInputStream(urlConn.getInputStream()); <- this method has been
depreacted with the BufferedReader

BufferedReader inStream = new BufferedReader(new
InputStreamReader(urlConn.getInputStream()));

StringBuffer lineBuf = new StringBuffer();
String line = null;

while ((line = inStream.readLine()) != null) {
    lineBuf.append(line);
}

pattern = Pattern.compile(prePattern);
matcher = pattern.matcher(lineBuf);

//System.out.println(lineBuf);

if(matcher.find()) { // results are complete

    out.println(HEADER_XML);

    printElement("sift",0);

    pattern = Pattern.compile(mainPattern);
    matcher = pattern.matcher(lineBuf);

    if(matcher.find()) {

        //printElement("RESULT", matcher.group(0),1);

        printElement("snp", matcher.group(1),1);
        printElement("amino_acid_change", matcher.group(2),1);
        printElement("protein", matcher.group(3),1);
        printElement("amino_acid_1", matcher.group(4),1);
        printElement("amino_acid_2", matcher.group(19),1);
```

```

        printElement("prediction_1_orthologue", matcher.group(6),1);
        printElement("score_1_orthologue", matcher.group(8),1);
        printElement("prediction_1_homologue", matcher.group(13),1);
        printElement("score_1_homologue", matcher.group(15),1);

        printElement("prediction_2_orthologue", matcher.group(21),1);
        printElement("score_2_orthologue", matcher.group(23),1);
        printElement("prediction_2_homologue", matcher.group(25),1);
        printElement("score_2_homologue", matcher.group(27),1);

        printElement("median_sequence_ic_orthologue", matcher.group(10),1);
        printElement("alignment_orthologue", matcher.group(11),1);
        printElement("median_sequence_ic_homologue", matcher.group(17),1);
        printElement("alignment_homologue", matcher.group(18),1);
    }

    printElement("/sift",0);

    //inStream.close();

    } //end of all if statements
} //end of try

catch (Exception e) {
    out.println(HEADER_XML);
    System.err.println(e);
} //end of catch

} //end of processRequest

public String constructQuery(Iterator queryIterator) {

    StringBuffer termBuf = new StringBuffer("");
    HashMap groupMap = new HashMap();
    String ttype = null;
    String tvalue = null;
    String field = null;
    String hexencoded = null;
    //String retQuery = null;

```

```

while (queryIterator.hasNext()) {
    // Retrieve the term values that were parsed from the query URL.
    HashMap queryMap = (HashMap) queryIterator.next();
    ttype = (String) queryMap.get(AttribOption.OPTION_PARAM_NAME);
    tvalue = (String) queryMap.get(SIFT_OPTION_MAP_TVALUE);
    field = (String) queryMap.get(SIFT_OPTION_MAP_FIELD);
    hexencoded = (String)
    queryMap.get(SIFT_OPTION_MAP_HEXENCODED);

    // Convert term values from hex to ascii if necessary.
    if (hexencoded != null && hexencoded.equals("true")) {
        tvalue = StringUtility.convertHexToASCII(tvalue);
    }

    //retQuery = field + "=rs" + tvalue;
    retQuery = tvalue;
}

//System.out.println ("This is the returned query: " + retQuery);
return retQuery;
} //end of constructQuery Method

```

```

public void printElement(String tag, String content, int tabs) {
    String tabStr = "";
    content = content.trim();
    for(int o = 0; o < tabs; o++)
        tabStr += "\t";
    //System.out.println(tabStr + "<" + tag + ">" + CDATA_START + content +
    CDATA_END + "</" + tag + ">");
    out.println(tabStr + "<" + tag + ">" + CDATA_START + content +
    CDATA_END + "</" + tag + ">");
}

```

```

public void printElement(String tag, int tabs) {
    String tabStr = "";
    for(int o = 0; o < tabs; o++)
        tabStr += "\t";
    //System.out.println(tabStr + "<" + tag + ">");
}

```

```

        out.println(tabStr + "<" + tag + ">");
    }

    /**
     * getUsage
     * @return
     */
    public String getUsage() {
        return(
            HEADER_XML + "\n" +
            "<" + TAG_WRAPPER_SOURCE + ">\n" +
            " <help>\n" +
            " <usage>\n" +
            " <![CDATA[ \n" + HELP_TEXT +
            " ]]>\n" +
            " </usage>\n" +
            " </help>\n" +
            "</" + TAG_WRAPPER_SOURCE + ">"
        );
    }

    public void terminate() {
        // TODO Auto-generated method stub

    }

}

/*
 *
 * Created on Sept 19, 2007
 *
 *
 * @author Terry Shen
 *
 *
 * TODO To change the template for this generated type comment go to
 * Window - Preferences - Java - Code Style - Code Templates

```

```

*
*
*/

package org.biomediator.wrappers;

import javax.servlet.http.Cookie;

import org.biomediator.util.AttribOption;
import org.biomediator.util.StringUtility;

import java.net.URL;

import java.lang.Object.*;
import java.lang.Class.*;

import java.io.*;
import java.net.*;
import java.util.*;
import java.util.regex.Matcher;
import java.util.regex.Pattern;

public class TFSearchWrapper extends AbstractWrapper {

    public static final String TAG_WRAPPER_SOURCE =
        "TFSEARCH_source";
    public static final String HELP_TEXT =
        "Here's some help stuff... ";

    public static final String TFSEARCH_HELP_TEXT =
        "    Description:\n" +
        "        Provides an interface for searching the TFSEARCH
database.\n" +
        "\n" +
        "    Usage:\n" +
        "    Servlet examples:\n" +
        "    Usage:\n" +

```

```

        " Servlet examples:\n" +
        "
?term=term[tvalue=rs405509,hexencoded=false,field=target]" +
        "
example=?term=term[tvalue=rs451061,hexencoded=false,field=target]";

/* rs630612
 * chr1:25569579-25569580
 * rs405509
 * 19, chromStart=50100675, chromEnd=50100676
 **/

//http://www.cbrc.jp/htbin/nph-
tfsearch?label=rs2545&seq=CTAATAGTGGAACCCTGAGACTTTAAAT
CTGCAAAGGGGTTTAATAATGCA&taxonomy=V&threshold=80

public static final String TFSEARCH_SOURCE_TAG =
"TFSEARCH_source";
public static final String TFSEARCH_CMD_TERM = "term";
public static final String TFSEARCH_OPTION_MAP_FIELD = "field";
public static final String TFSEARCH_OPTION_MAP_TVALUE = "tvalue";
public static final String TFSEARCH_OPTION_MAP_HEXENCODED =
"hexencoded";

public static final String CDATA_START = "<![CDATA[ ";
public static final String CDATA_END = " ]]>";
public static final String ENDLINE = " ";

public static final String urlBaseXML = "http://www.cbrc.jp/htbin/nph-
tfsearch?taxonomy=V&threshold=60&label=label&";

public static final String prePatternTFSEARCH = "TFMATRIX entries with
High-scoring:(.*)Total";
public static final String TFSEARCHPattern = "([\s\\&lt;\\&gt;]+?)<A
HREF=http://www.cbrc.jp/htbin/bget_tfmatrix\\?\\w+>\\w+?</A>
([^\u2000]*?)\\b+([0-9][0-9]\\.[0-9])";
public static final String TFSEARCHSequence = "High-scoring: 1 ([A-
Z\\s]+) entry";

```

```

    public String baseURL =
        "http://www.cbrc.jp/research/db/TFSEARCH.html";
    public String maincookie="";
    public String label = "rs1447295";
    public String taxonomy = "V";
    public String threshold = "60";

    public String proxyUrl = null;
    private Pattern pattern = null;
    private Matcher matcher = null;
    private HashMap queryMap = null;

    private String ttype = null;
    private String tvalue = null;
    private String field = null;
    private String hexencoded = null;

    private String convertedQuery = null;
    private String retQuery = null;

    public static void main(String[] args) {

        TFSearchWrapper TFSEARCH = new TFSearchWrapper();
            //invoking the constructor
        TFSEARCH.out = new PrintWriter(System.out, true);
        if (args.length > 0) {
            TFSEARCH.processRequest(args[0]);
        }
        else {
            TFSEARCH.processRequest("");
        }
    }

    public void processRequest(String query) {

        // Make sure the servlet is initialized.
        initializeServlet(TFSEARCH_HELP_TEXT, TFSEARCH_SOURCE_TAG,
            getWrapperProperties("TRANSFAC"));
    }

```

```

// Parse the query string and process commands.
if (query == null || query.equals("")) {
    query = CMD_HELP;
}
AttribOption ao = new AttribOption(query);
processCommands(ao, getWrapperProperties("transfac"));

// Check to see if we should exit immediately.
if (isExitCmd) {
    return;
}

String proxy =
    wrapperProperties.getProperty(WrapperProperties.PROP_PROXY_URL);
if (proxy == null) {
    proxyUrl = "";
}
else {
    proxyUrl = proxy + "?";
}

TreeSet termTypeSet = ao.getAttribMapSet(TFSEARCH_CMD_TERM);

if (termTypeSet != null) {

    convertedQuery = constructQuery(termTypeSet.iterator());
    //out.println("The convertedquery is: " + convertedQuery);

    retQuery = field + "=" + convertedQuery ;
    //out.println("The retQuery is: " + retQuery);
}

try {

    // Download Original XML page.
    String queryURLXML = proxyUrl + urlBaseXML + retQuery;
    BufferedReader brXML = new BufferedReader(new InputStreamReader(new
    URL(queryURLXML).openStream()));
    StringBuffer lineBufXML = new StringBuffer();

```

```

String lineXML = null;

//out.println("The returned XML output URL is: " + queryURLXML);

while ((lineXML = brXML.readLine()) != null) {
    lineBufXML.append(lineXML);
}

String dataXML = lineBufXML.toString();

//out.println("This is the dataXML: "+ dataXML);

out.println(HEADER_XML);

printElement("tfsearch_source",0);

out.println("\t<tfsearch_id>" + CDATA_START + convertedQuery +
CDATA_END + "</tfsearch_id>");

pattern = Pattern.compile(TFSEARCHSequence);
matcher = pattern.matcher(dataXML);
while(matcher.find()) {
    printElement("tfsearch_sequence", matcher.group(1),1);
}

pattern = Pattern.compile(prePatternTFSEARCH);
matcher = pattern.matcher(dataXML);
//out.println("This is the pattern: "+ pattern);
//out.println("This is the matcher: "+ matcher);

if (matcher.find()) { // results are complete

    pattern = Pattern.compile(TFSEARCHPattern);
    matcher = pattern.matcher(dataXML);
    //out.println("This is the pattern: "+ pattern);
    //out.println("This is the matcher: "+ matcher);
    while(matcher.find()) {
        //printElement("tfsearch_record", matcher.group(1)+ " " +
matcher.group(2),1);
        printElement("tfrecord",0);
        String arrows = matcher.group(1);

```

```

        arrows = arrows.replaceAll("\\s", ".");
        arrows = arrows.replaceAll("&lt;", "<");
        arrows = arrows.replaceAll("-", "=");
        out.println("\t<tfsearch_motif_pattern>" + CDATA_START +
arrows + CDATA_END + "</tfsearch_motif_pattern>");
        //printElement("tfsearch_motif_pattern", matcher.group(1),1);
        printElement("tfsearch_record_name", matcher.group(2),1);
        printElement("tfsearch_record_score", matcher.group(3),1);
        printElement("/tfrecord",0);
    }

        printElement("/tfsearch_source",0);
    } //end of all if statements

    brXML.close();

} //end of try

catch (Exception e) {

    out.println(HEADER_XML);
    System.err.println(e.getClass() + ": " + e.getMessage());
    e.printStackTrace();

} //end of catch

} //end of processRequest

public String constructQuery(Iterator queryIterator) {

    StringBuffer termBuf = new StringBuffer("");
    HashMap groupMap = new HashMap();

    String retID = null;

    //out.println("This is the queryIterator: " + queryIterator);

    if (queryIterator.hasNext()) {

```

```

while (queryIterator.hasNext()) {
    // Retrieve the term values that were parsed from the query URL.
    HashMap queryMap = (HashMap) queryIterator.next();
    ttype = (String) queryMap.get(AttribOption.OPTION_PARAM_NAME);
    tvalue = (String) queryMap.get(TFSEARCH_OPTION_MAP_TVALUE);
    field = (String) queryMap.get(TFSEARCH_OPTION_MAP_FIELD);
    hexencoded = (String)
queryMap.get(TFSEARCH_OPTION_MAP_HEXENCODED);

        //out.println("String retID is: " + retID);

    // Convert term values from hex to ascii if necessary.
    if (hexencoded != null && hexencoded.equals("true")) {
        tvalue = StringUtility.convertHexToASCII(tvalue);

        retID = tvalue;
    }

    else {

        retID = tvalue;
    }

}
}

//out.println("This is the retID value: " + retID);

return retID;
}

```

```

public String getUsage() {
    return(
        HEADER_XML + "\n" +
        "<" + TAG_WRAPPER_SOURCE + ">\n" +
        " <help>\n" +
        " <usage>\n" +
        " <![CDATA[ \n" + HELP_TEXT +
        " ]]>\n" +

```

```

        " </usage>\n" +
        " </help>\n" +
        "</" + TAG_WRAPPER_SOURCE + ">"
    );
}

public void printElement(String tag, String content, int tabs) {
    String tabStr = "";
    content = content.trim();
    for(int o = 0; o < tabs; o++)
        tabStr += "\t";
    out.println(tabStr + "<" + tag + ">" + CDATA_START + content +
        CDATA_END + "</" + tag + ">");
}

public void printElement(String tag, int tabs) {
    String tabStr = "";
    for(int o = 0; o < tabs; o++)
        tabStr += "\t";
    out.println(tabStr + "<" + tag + ">");
}

public void terminate() {
    // TODO Auto-generated method stub
}

}

/*
 *
 * Created on Jul 24, 2006
 *
 *
 * @author Terry Shen, see all the other code
 *
 * This wrapper uses HashMap to capture the query string.

```

```

* This wrapper also has a repeating pattern!
*
*
* TODO To change the template for this generated type comment go to
* Window - Preferences - Java - Code Style - Code Templates
*
*
*/

```

```
package org.biomediator.wrappers;
```

```

import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.io.PrintWriter;
import java.net.URL;
import java.net.URLDecoder;
import java.util.HashMap;
import java.util.Iterator;
import java.util.TreeSet;
//import java.util.TreeSet;
import java.util.regex.Matcher;
import java.util.regex.Pattern;

```

```

import org.biomediator.util.AttribOption;
import org.biomediator.util.StringUtility;

```

```
//import org.biomediator.util.AttribOption;
```

```
//import com.oroenc.text.regex.Perl5StreamInput;
```

```
public class UCSCEvolWrapper extends AbstractWrapper {
```

```

    public static final String UCSCEVOL_HELP_TEXT
        = "      Description:\n" +
        "      Provides an interface for searching the UCSC Evolutionary
        Conservation (PhastCons) database.\n" +
        "\n" +
        "      Usage:\n" +
        "      Servlet examples:\n" +

```

```

"                ?term=term[tvalue=chr1:2107228-
2107270,hexencoded=false,field=position,db=db,dbvalue=hg18]";
//                chr1:2108399-2108399,rs2503699 is an example that
has a score
//                chr1:167778357-
167778358,rs4525 is an example that doesn't work!
##bin chrom chromStart chromEnd name score strand refNCBI
refUCSC observed molType class valid avHet
avHetSE func locType weight
//1865 chr1 167778357 167778358 rs4525 0 - A T
A/G genomic single by-cluster,by-frequency,by-2hit-2allele
0.34449

public static final String UCSCEVOL_SOURCE_TAG = "ucscevol_source";
public static final String UCSCEVOL_CMD_TERM = "term";
public static final String UCSCEVOL_OPTION_MAP_FIELD = "field";
public static final String UCSCEVOL_OPTION_MAP_TVALUE = "tvalue";
public static final String UCSCEVOL_OPTION_MAP_DB = "db";
public static final String UCSCEVOL_OPTION_MAP_DBVALUE =
"dbvalue";
public static final String UCSCEVOL_OPTION_MAP_HEXENCODED =
"hexencoded";

public static final String CDATA_START = "<![CDATA[ ";
public static final String CDATA_END = " ]]>";
public static final String ENDLINE = " ";

public static final String urlBase = "http://genome.ucsc.edu/cgi-
bin/hgTables?hgta_compressType=none&hgta_group=compGeno&hgta_out
putType=wigData&hgta_regionType=range&hgta_table=phastCons17way&
hgta_track=multiz17way&org=Human&submit=submit&hgta_doTopSubmit
=1&";

//public static final String urlQuery = "&db=hg17&";
//hg18 is the newest build - May 2006

// &position=chr1:2107228-2107230

/*
//This pattern is for version hg 17

```

```

public static final String prePattern = "track type=wiggle_0
name=\"Conservation\" description=\"Vertebrate Multiz Alignment &
Conservation\"(.*?)";
public static final String positionIDPattern = "chrom
specified:\\s(+)\\s#\\sposition specified:\\s(+)";
public static final String phastconPattern = "(\\d+)\\s+(\\d+\\.\\d+)"; // \\s
equals spaces
*/

//This pattern is for version hg 18
public static final String prePattern = "track type=wiggle_0 name=\"17-Way
Cons\" description=\"Vertebrate Multiz Alignment &
Conservation([<math>^</math><math>◇</math>]*)";
public static final String positionIDPattern = "chrom
specified:\\s+(.+?)STOP#\\s+position specified:\\s+(.+?)STOP#";
//chrom specified:\\s(+)STOP#\\sposition specified:\\s(+)
//chrom specified:\\s+(.+?)STOP#\\s+position specified:\\s+(\\d+-\\d+)
public static final String phastconPattern =
"STOP(\\d+)\\s+(\\d+\\.\\d+\\.\\d+)STOP"; // \\s equals spaces
//(\\d+)\\s+(\\d+\\.\\d+)

/*
 * NEWEST VERSION - UGH - Oct 2007
 *
 * track type=wiggle_0 name="17-Way Cons" description="Vertebrate
Multiz Alignment & Conservation (17 Species)"
# output date: 2007-10-13 19:45:44 UTC
# chrom specified: chr1
# position specified: 2107228-2107270
# This data has been compressed with a minor loss in resolution.
# (Worst case: 0.0078125) The original source data
# (before querying and compression) is available at
# http://hgdownload.cse.ucsc.edu/downloads.html
variableStep chrom=chr1 span=1
 *
 */

public String proxyUrl = null;
private Pattern pattern = null;

```

```

private Matcher matcher = null;
private HashMap queryMap = null;

private String ttype = null;
private String tvalue = null;
private String field = null;
private String hexencoded = null;
private String db = null;
private String dbvalue = null;
private String retQuery = null;

/**
 * main
 * @param args
 */
public static void main(String[] args) {
    UCSCEvolWrapper w = new UCSCEvolWrapper();
    //invoking the constructor
    w.out = new PrintWriter(System.out, true);
    if (args.length > 0) {
        w.processRequest(args[0]);
    }
    else {
        w.processRequest("");
    }
} //end of main method

/**
 * processRequest
 * @param query
 */
public void processRequest(String query) {
    initializeServlet(UCSCEVOL_HELP_TEXT,
UCSCEVOL_SOURCE_TAG, getWrapperProperties("ucscevol"));
    if (query == null || query.equals("")) {
        query = CMD_HELP;
    }

    // Check to see if we should exit immediately.

```

```

        if (isExitCmd) {
            return;
        }

        String proxy =
wrapperProperties.getProperty(WrapperProperties.PROP_PROXY_URL);

        doQuery(query);

    } //end of ProcessRequest method

```

```

public void doQuery(String query) {

   _ATTRIB_OPTION ao = new_ATTRIB_OPTION(query);
    processCommands(ao, getWrapperProperties("ucscevol"));

    String proxy =
wrapperProperties.getProperty(WrapperProperties.PROP_PROXY_URL);
    if (proxy == null) {
        proxyUrl = "";
    }
    else {
        proxyUrl = proxy + "?";
    }

    String convertedQuery = null;
    TreeSet termTypeSet =
ao.getAttribMapSet(UCSCEVOL_CMD_TERM);
    if (termTypeSet != null) {
        convertedQuery = constructQuery(termTypeSet.iterator());
    }

    try {

        // Download the URL.
        String url = proxyUrl + urlBase + convertedQuery;

        //System.out.println("This is the URL: " + url);

```

```

// Download HTML page.
BufferedReader br = new BufferedReader(new
InputStreamReader(new URL(url).openStream()));
StringBuffer lineBuf = new StringBuffer();
String line = null;

while ((line = br.readLine()) != null) {
    //lineBuf.append(line + '\n');           //
'\n' means append to next line
    lineBuf.append(line + "STOP");
}

//System.out.println("This is the lineBuf:" + lineBuf);
//System.out.println("This is the prePattern:" + prePattern);

pattern = Pattern.compile(prePattern);
matcher = pattern.matcher(lineBuf);

if (matcher.find()) { // results are complete

    out.println(HEADER_XML);
    printElement("evolutionary_conservation_group",0);
    out.println("\t<source_id>" + CDATA_START +
tvalue + CDATA_END + "</source_id>");

    Pattern pattern1 = Pattern.compile(phastconPattern);
    Matcher matcher1 = pattern1.matcher(lineBuf);
    while(matcher1.find()){
        String comparescore = (matcher1.group(2));

        /* NOT SURE IF THIS PART IS
NECESSARY. CAUSES MULTIPLE HEADER XMLS
        if(comparescore.equals("0")) {
            out.println(HEADER_XML);
        }
        */

        //if {

```

```

        printElement("evolutionary_conservation",2);
                                printElement("chrom_location",
matcher1.group(1),3);

                                printElement("phastcon_score",
matcher1.group(2),3);

        printElement("/evolutionary_conservation",2);

                                //} //end of else

                                } //end of while

        printElement("/evolutionary_conservation_group",0);

                                } //end of if
    } //end of try

        catch (Exception e) {

                out.println(HEADER_XML);
                System.err.println(e.getClass() + ": " +
e.getMessage());
                e.printStackTrace();

        } //end of catch

    } //end of processRequest

public String constructQuery(Iterator queryIterator) {

        StringBuffer termBuf = new StringBuffer("");

```

```

HashMap groupMap = new HashMap();
//String ttype = null;
//String tvalue = null;
//String field = null;
//String hexencoded = null;
//String retQuery = null;
//String db = null;
//String dbvalue = null;

//Example of what the parameter result should be:
//term=term[tvalue=chr1:2107228-
2107230,hexencoded=false,field=position] ;

while (queryIterator.hasNext()) {
    // Retrieve the term values that were parsed from the
query URL.
    HashMap queryMap = (HashMap)
queryIterator.next();
    ttype = (String)
queryMap.get(AttribOption.OPTION_PARAM_NAME);
    tvalue = (String)
queryMap.get(UCSCEVOL_OPTION_MAP_TVALUE);
    field = (String)
queryMap.get(UCSCEVOL_OPTION_MAP_FIELD);
    db = (String)
queryMap.get(UCSCEVOL_OPTION_MAP_DB);
    dbvalue = (String)
queryMap.get(UCSCEVOL_OPTION_MAP_DBVALUE);

    hexencoded = (String)
queryMap.get(UCSCEVOL_OPTION_MAP_HEXENCODED);

    // Convert term values from hex to ascii if necessary.
    if (hexencoded != null && hexencoded.equals("true"))
    {
        tvalue =
StringUtility.convertHexToASCII(tvalue);
        //dbvalue =
StringUtility.convertHexToASCII(dbvalue);
    }
}

```

```

        //retQuery = field + "=rs" + tvalue + "&" + db + "=" +
dbvalue+ "&" + sz + "=" + szvalue; //if you wanted to take out the rs
        retQuery = field + "=" + tvalue + "&" + db + "=" +
dbvalue;
    }

    //System.out.println ("This is the returned query: " +
retQuery);
    return retQuery;
} //end of constructQuery Method

/**
 * getUsage
 * @return
 */
public String getUsage() {
    return(
        HEADER_XML + "\n" +
        "<" + TAG_WRAPPER_SOURCE + ">\n" +
        " <help>\n" +
        " <usage>\n" +
        " <![CDATA[ \n" + HELP_TEXT +
        " ]]>\n" +
        " </usage>\n" +
        " </help>\n" +
        "</" + TAG_WRAPPER_SOURCE + ">"
    );
}

public void printElement(String tag, String content, int tabs) {
    String tabStr = "";
    content = content.trim();
    for(int o = 0; o < tabs; o++)
        tabStr += "\t";
    out.println(tabStr + "<" + tag + ">" + CDATA_START +
content + CDATA_END + "</" + tag + ">");
}

public void printElement(String tag, int tabs) {

```

```
        String tabStr = "";
        for(int o = 0; o < tabs; o++)
            tabStr += "\t";
        out.println(tabStr + "<" + tag + ">");
    }

    public void terminate() {
        // TODO Auto-generated method stub

    }

}

/*
 *
 * Created on Jul 25, 2006
 *
 *
 * @author Terry Shen, see all the other code
 *
 * This wrapper uses HashMap to capture the query string.
 *
 *
 * TODO To change the template for this generated type comment go to
 * Window - Preferences - Java - Code Style - Code Templates
 *
 *
 */

package org.biomediator.wrappers;

import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.io.PrintWriter;
import java.net.URL;
import java.net.URLDecoder;
import java.util.HashMap;
import java.util.Iterator;
```

```

import java.util.StringTokenizer;
import java.util.TreeSet;
//import java.util.TreeSet;
import java.util.regex.Matcher;
import java.util.regex.Pattern;

import org.biomediator.util.AttribOption;
import org.biomediator.util.StringUtility;

//import org.biomediator.util.AttribOption;

//import com.oroinc.text.regex.Perl5StreamInput;

public class UCSCGenePredWrapper extends AbstractWrapper {

    public static final String UCSCGENEPRED_HELP_TEXT
        = "      Description:\n" +
        "      Provides an interface for searching the UCSC Genscan database.\n" +
        "\n" +
        "      Usage:\n" +
        "      Servlet examples:\n" +
        "      ?term=term[tvalue=chr1:2107228-
        2117230,hexencoded=false,field=position,db=db,dbvalue=hg18]" ;
        //chr1:2064917-
        2064937, rs451061
        //chr19:50100675-
        50100676 = doesn't return info

    public static final String UCSCGENEPRED_SOURCE_TAG =
        "ucsc_gene_pred_source";
    public static final String UCSCGENEPRED_CMD_TERM = "term";
    public static final String UCSCGENEPRED_OPTION_MAP_FIELD =
        "field";
    public static final String UCSCGENEPRED_OPTION_MAP_TVALUE =
        "tvalue";
    public static final String UCSCGENEPRED_OPTION_MAP_DB = "db";

```



```

public static final String mainPattern = "(NT_+)" +
    "(.)" +
    "(.)" +
    "(.)" +
    "(.)" +
    "(.)" +
    "(.)" +
    "(.)" +
    "(.)" +
    "(.)";

```

```

public String proxyUrl = null;
private Pattern pattern = null;
private Matcher matcher = null;
private HashMap queryMap = null;

```

```

private String ttype = null;
private String tvalue = null;
private String field = null;
private String hexencoded = null;
private String db = null;
private String dbvalue = null;

```

```

private String convertedQuery = null;
private String retQuery = null;

```

```

/**
 * main
 * @param args
 */

```

```

public static void main(String[] args) {
    UCSCGenePredWrapper w = new UCSCGenePredWrapper();
        //invoking the constructor
    w.out = new PrintWriter(System.out, true);
    if (args.length > 0) {
        w.processRequest(args[0]);
    }
    else {
        w.processRequest("");
    }
} //end of main method

/**
 * processRequest
 * @param query
 */
public void processRequest(String query) {
    //queryMap = buildQueryMap(query); (old version)
    // Make sure the servlet is initialized.
    initializeServlet(UCSCGENEPRED_HELP_TEXT,
        UCSCGENEPRED_SOURCE_TAG,
        getWrapperProperties("ucscgenepred"));
    // Parse the query string and process commands.
    if (query == null || query.equals("")) {
        query = CMD_HELP;
    }

    // Check to see if we should exit immediately.
    if (isExitCmd) {
        return;
    }

    String proxy =
        wrapperProperties.getProperty(WrapperProperties.PROP_PROXY_URL);

    doQuery(query);
} //end of ProcessRequest method

```

```

public void doQuery(String query) {

    AttribOption ao = new AttribOption(query);
    processCommands(ao, getWrapperProperties("ucscgenepred"));

    String proxy =
        wrapperProperties.getProperty(WrapperProperties.PROP_PROXY_URL);
    if (proxy == null) {
        proxyUrl = "";
    }
    else {
        proxyUrl = proxy + "?";
    }

    //String convertedQuery = null;
    TreeSet termTypeSet =
        ao.getAttribMapSet(UCSCGENEPRED_CMD_TERM);

    //System.out.println ("This is the returned query: " + retQuery);

    if (termTypeSet != null) {
        convertedQuery = constructQuery(termTypeSet.iterator());
        retQuery = field + "=" + tvalue + "&" + db + "=" + dbvalue;
    }

    //System.out.println("original query: " + query);
    //System.out.println("converted query: " + convertedQuery);

    //This is the part where I try and convert the chromosome location because the
    //source does not return the Source/ID.
    //Didn't work, tried another method.
    /*
    queryMap = buildQueryMap(query);
    //System.out.println("This is the queryMap: " + queryMap);
    String returnedquery = ((String[])queryMap.get("tvalue"))[0];
    System.out.println("This is the returnedquery: " + returnedquery);
    */
}

```

```

try {

// Download the URL.
String url = proxyUrl + urlBase + retQuery;
    //String url = urlBase + retQuery;

//Lots of print outs to test and look at variables and such.
/*
System.out.println("urlBase: " + urlBase);
System.out.println("convertedQuery: " + convertedQuery);
System.out.println("retQuery: " + retQuery);
System.out.println("This is the URL: " + url);
*/

        // Download HTML page.
        BufferedReader br = new BufferedReader(new
InputStreamReader(new URL(url).openStream()));
        StringBuffer lineBuf = new StringBuffer();
        String line = null;

while ((line = br.readLine()) != null) {
    lineBuf.append(line + '\n');                // '\n' means append to
next line
}

//System.out.println("LINEBUF: " + lineBuf);
//System.out.println("PREPATTERN: " + prePattern);

pattern = Pattern.compile(prePattern);
matcher = pattern.matcher(lineBuf);

out.println(HEADER_XML);
printElement("genscan_prediction",0);
out.println("\t<source_id>" + CDATA_START + convertedQuery +
CDATA_END + "</source_id>");

if (matcher.find()) { // results are complete

        pattern = Pattern.compile(mainPattern);
        matcher = pattern.matcher(lineBuf);

```

```

if(matcher.find()) {

    printElement("name", matcher.group(1),1);
    printElement("chrom", matcher.group(2),1);
    printElement("strand", matcher.group(3),1);
    printElement("tx_Start", matcher.group(4),1);
    printElement("tx_End", matcher.group(5),1);
    printElement("cds_Start", matcher.group(6),1);
    printElement("cds_End", matcher.group(7),1);
    printElement("exon_Count", matcher.group(8),1);
    String exStart = matcher.group(9);
    String exEnd = matcher.group(10);
    exStart = exStart.replaceAll("\\\\", "\\");
    exEnd = exEnd.replaceAll("\\\\", "\\");
    printElement("exon_Starts", exStart,1);
    printElement("exon_Ends", exEnd,1);
}

    printElement("/genscan_prediction",0);

} //end of all if statements

br.close();

} //end of try

catch (Exception e) {

    out.println(HEADER_XML);
    System.err.println(e.getClass() + ": " + e.getMessage());
    e.printStackTrace();

} //end of catch

} //end of processRequest

public String constructQuery(Iterator queryIterator) {

    StringBuffer termBuf = new StringBuffer("");

```

```

HashMap groupMap = new HashMap();
/*
String ttype = null;
String tvalue = null;
String field = null;
String hexencoded = null;
String retQuery = null;
String db = null;
String dbvalue = null;
*/

while (queryIterator.hasNext()) {
    // Retrieve the term values that were parsed from the query URL.
    HashMap queryMap = (HashMap) queryIterator.next();
    ttype = (String) queryMap.get(AttribOption.OPTION_PARAM_NAME);
    tvalue = (String)
        queryMap.get(UCSCGENEPRED_OPTION_MAP_TVALUE);
    field = (String) queryMap.get(UCSCGENEPRED_OPTION_MAP_FIELD);
    db = (String) queryMap.get(UCSCGENEPRED_OPTION_MAP_DB);
    dbvalue = (String)
        queryMap.get(UCSCGENEPRED_OPTION_MAP_DBVALUE);

    hexencoded = (String)
        queryMap.get(UCSCGENEPRED_OPTION_MAP_HEXENCODED);

    // Convert term values from hex to ascii if necessary.
    if (hexencoded != null && hexencoded.equals("true")) {
        tvalue = StringUtility.convertHexToASCII(tvalue);
        //dbvalue = StringUtility.convertHexToASCII(dbvalue);
    }

    //retQuery = field + "=" + tvalue + "&" + db + "=" + dbvalue;
}

//System.out.println ("This is the returned query: " + retQuery);
//return retQuery;
return tvalue;
} //end of constructQuery Method

```

```

public HashMap buildQueryMap (String convertedQuery) {
    //System.out.println(" decoded converted query: "+ convertedQuery);
    HashMap queryMap = new HashMap();
    StringTokenizer st = new StringTokenizer(convertedQuery, "&");
    while (st.hasMoreTokens())
    {
        StringTokenizer st1 = new StringTokenizer(st.nextToken(),
"=");

        String key = st1.nextToken();
        String value = (st1.hasMoreTokens()?st1.nextToken(): "");
        //System.out.println(" key: "+key+", value: "+value);
        queryMap.put(key, new String[]{value});
        //System.out.println(" key: "+key+", value: "+value);
    }
    return queryMap;
}

/**
 * getUsage
 * @return
 */
public String getUsage() {
    return(
        HEADER_XML + "\n" +
        "<" + TAG_WRAPPER_SOURCE + ">\n" +
        " <help>\n" +
        " <usage>\n" +
        " <![CDATA[ \n" + HELP_TEXT +
        " ]]>\n" +
        " </usage>\n" +
        " </help>\n" +
        "</" + TAG_WRAPPER_SOURCE + ">"
    );
}

public void printElement(String tag, String content, int tabs) {
    String tabStr = "";
    content = content.trim();
    for(int o = 0; o < tabs; o++)

```

```

        tabStr += "\t";
        out.println(tabStr + "<" + tag + ">" + CDATA_START + content +
            CDATA_END + "</" + tag + ">");
    }

    public void printElement(String tag, int tabs) {
        String tabStr = "";
        for(int o = 0; o < tabs; o++)
            tabStr += "\t";
        out.println(tabStr + "<" + tag + ">");
    }

    public void terminate() {
        // TODO Auto-generated method stub
    }
}

/*
 *
 * Created on Jul 25, 2006
 *
 *
 * @author Terry Shen, see all the other code
 *
 * This wrapper uses HashMap to capture the query string.
 *
 *
 * TODO To change the template for this generated type comment go to
 * Window - Preferences - Java - Code Style - Code Templates
 *
 *
 */

package org.biomediator.wrappers;

import java.io.BufferedReader;

```

```

import java.io.InputStreamReader;
import java.io.PrintWriter;
import java.net.URL;
import java.net.URLDecoder;
import java.util.HashMap;
import java.util.Iterator;
import java.util.StringTokenizer;
import java.util.TreeSet;
//import java.util.TreeSet;
import java.util.regex.Matcher;
import java.util.regex.Pattern;

import org.biomediator.util.AttribOption;
import org.biomediator.util.StringUtility;

//import org.biomediator.util.AttribOption;

//import com.oroenc.text.regex.Perl5StreamInput;

public class UCSCTissueExpWrapper extends AbstractWrapper {

    public static final String UCSCTISSUEEXP_HELP_TEXT
        = "      Description:\n" +
        "      Provides an interface for searching the UCSC GNFAtlas2
        database.\n" +
        "\n" +
        "      Usage:\n" +
        "      Servlet examples:\n" +
        "      ?term=term[tvalue=chr1:2064678-
        2066178,hexencoded=false,field=position,db=db,dbvalue=hg18]" ;
        //rs451061

    public static final String UCSCTISSUEEXP_SOURCE_TAG =
        "ucsc_tissue_exp_source";
    public static final String UCSCTISSUEEXP_CMD_TERM = "term";
    public static final String UCSCTISSUEEXP_OPTION_MAP_FIELD =
        "field";
    public static final String UCSCTISSUEEXP_OPTION_MAP_TVALUE =
        "tvalue";
    public static final String UCSCTISSUEEXP_OPTION_MAP_DB = "db";

```

```

public static final String UCSCTISSUEEXP_OPTION_MAP_DBVALUE =
"dbvalue";
public static final String
UCSCTISSUEEXP_OPTION_MAP_HEXENCODED = "hexencoded";

public static final String CDATA_START      = "<![CDATA[ ";
public static final String CDATA_END        = " ]]>";
public static final String ENDLINE          = " ";

public static final String urlBase = "http://genome.ucsc.edu/cgi-
bin/hgTables?hgta_compressType=none&" +
"hgta_database=hg18&hgta_fieldSelectTable=hg18.gnfAtlas2&hgta_fs.chec
k.hg18.gnfAtlas2.bin=0&hgta_fs.check.hg18.gnfAtlas2.blockCount=0&" +
"hgta_fs.check.hg18.gnfAtlas2.blockSizes=0&hgta_fs.check.hg18.gnfAtlas2.
chrom=1&hgta_fs.check.hg18.gnfAtlas2.chromEnd=1&" +
"hgta_fs.check.hg18.gnfAtlas2.chromStart=1&hgta_fs.check.hg18.gnfAtlas2.
chromStarts=1&hgta_fs.check.hg18.gnfAtlas2.expCount=1&" +
"hgta_fs.check.hg18.gnfAtlas2.expIds=1&hgta_fs.check.hg18.gnfAtlas2.exp
Scores=1&hgta_fs.check.hg18.gnfAtlas2.name=1&" +
"hgta_fs.check.hg18.gnfAtlas2.reserved=1&hgta_fs.check.hg18.gnfAtlas2.sc
ore=1&hgta_fs.check.hg18.gnfAtlas2.strand=1&" +
"hgta_fs.check.hg18.gnfAtlas2.thickEnd=0&hgta_fs.check.hg18.gnfAtlas2.th
ickStart=0&hgta_fs.check.hgFixed.gladHumESOtherData.hVal=0&" +
"hgta_fs.check.hgFixed.gladHumESOtherData.name=0&hgta_fs.check.hgFixe
d.gladHumESOtherData.qVal=0&hgta_fs.check.hgFixed.gladHumESOther
Data.tissueQ=1&" +
"hgta_fs.linked.hg16.affyGnf1h=0&hgta_fs.linked.hg16.gnfAtlas2=0&hgta_f
s.linked.hg16.knownToGnfAtlas2=0&hgta_fs.linked.hg17.affyGnf1h=0&" +
"hgta_fs.linked.hg17.gnfAtlas2=0&hgta_fs.linked.hg17.knownToGnfAtlas2=
0&hgta_fs.linked.hg18.affyGnf1h=0&hgta_fs.linked.hg18.knownToGnfAtlas
2=0&" +
"hgta_fs.linked.hgFixed.gladHumES=0&hgta_fs.linked.hgFixed.gladHumES
OtherData=1&hgta_fs.linked.hgFixed.gladHumESRatio=0&" +
"hgta_fs.linked.hgFixed.gnfHumanAtlas2All=0&hgta_fs.linked.hgFixed.gnf
HumanAtlas2AllRatio=0&hgta_fs.linked.hgFixed.gnfHumanAtlas2Median=
0&" +
"hgta_fs.linked.hgFixed.gnfHumanAtlas2MedianExps=0&hgta_fs.linked.hgF
ixed.gnfHumanAtlas2MedianRatio=0&hgta_group=regulation&hgta_outFile
Name=&" +

```



```

"(-{0,1}+\\d+\\.\\.d+),+?\\t*" + "(-{0,1}+\\d+\\.\\.d+),+?\\t*" + "(-
{0,1}+\\d+\\.\\.d+),+?\\t*" + "(-{0,1}+\\d+\\.\\.d+),+?\\t*" + "(-
{0,1}+\\d+\\.\\.d+),+?\\t*" +
"(-{0,1}+\\d+\\.\\.d+),+?\\t*" + "(-{0,1}+\\d+\\.\\.d+),+?\\t*" + "(-
{0,1}+\\d+\\.\\.d+),+?\\t*" + "(-{0,1}+\\d+\\.\\.d+),+?\\t*" + "(-
{0,1}+\\d+\\.\\.d+),+?\\t*" +
"(-{0,1}+\\d+\\.\\.d+),+?\\t*" + "(-{0,1}+\\d+\\.\\.d+),+?\\t*" + "(-
{0,1}+\\d+\\.\\.d+),+?\\t*" + "(-{0,1}+\\d+\\.\\.d+),+?\\t*" +
"(.+)";
/*
"(\S+?\\s)*" +
".*?\\s*" +
"((-{0,1}+\\d+\\.\\.d+),)*\\s*" +
"([a-zA-Z]+)";
*/

public String proxyUrl = null;
private Pattern pattern = null;
private Matcher matcher = null;
private HashMap queryMap = null;

private String ttype = null;
private String tvalue = null;
private String field = null;
private String hexencoded = null;
private String db = null;
private String dbvalue = null;

private String convertedQuery = null;
private String retQuery = null;

/**
 * main
 * @param args
 */
public static void main(String[] args) {
    UCSTissueExpWrapper w = new UCSTissueExpWrapper();
    //invoking the constructor
    w.out = new PrintWriter(System.out, true);

```

```

        if (args.length > 0) {
            w.processRequest(args[0]);
        }
        else {
            w.processRequest("");
        }
    } //end of main method

/**
 * processRequest
 * @param query
 */
public void processRequest(String query) {
    //queryMap = buildQueryMap(query); (old version)
    // Make sure the servlet is initialized.
    initializeServlet(UCSCTISSUEEXP_HELP_TEXT,
UCSCTISSUEEXP_SOURCE_TAG,
getWrapperProperties("ucsctissueexp"));
    // Parse the query string and process commands.
    if (query == null || query.equals("")) {
        query = CMD_HELP;
    }

    // Check to see if we should exit immediately.
    if (isExitCmd) {
        return;
    }

    String proxy =
wrapperProperties.getProperty(WrapperProperties.PROP_PROXY_URL);

    doQuery(query);
} //end of ProcessRequest method

public void doQuery(String query) {

    AttribOption ao = new AttribOption(query);

```

```

processCommands(ao, getWrapperProperties("ucsctissueexp"));

String proxy =
wrapperProperties.getProperty(WrapperProperties.PROP_PROXY_URL);
if (proxy == null) {
    proxyUrl = "";
}
else {
    proxyUrl = proxy + "?";
}

//String convertedQuery = null;
TreeSet termTypeSet =
ao.getAttribMapSet(UCSCTISSUEEXP_CMD_TERM);

//System.out.println ("This is the returned query: " + retQuery);

if (termTypeSet != null) {
    convertedQuery = constructQuery(termTypeSet.iterator());
    retQuery = field + "=" + tvalue + "&" + db + "=" + dbvalue;
}

try {

    // Download the URL.
    String url = proxyUrl + urlBase + retQuery;

    //Lots of print outs to test and look at variables and such.
    /*
    System.out.println("urlBase: " + urlBase);
    System.out.println("convertedQuery: " + convertedQuery);
    System.out.println("retQuery: " + retQuery);
    */
    //System.out.println("This is the URL: " + url);
    //System.out.println("This is the URL for the tissue names: " +
urlBaseTissueName);

    // Download HTML page.

```

```

        BufferedReader br = new BufferedReader(new
InputStreamReader(new URL(url).openStream()));
        StringBuffer lineBuf = new StringBuffer();
        String line = null;

        while ((line = br.readLine()) != null) {
            lineBuf.append(line + '\t');           // '\n'
means append to next line
            //lineBuf.append(line);
        }

        //System.out.println("LINEBUF IS THIS WORKING?: " +
lineBuf);
        //System.out.println("PREPATTERN: " + prePattern);
        //System.out.println("MAINPATTERN: " + mainPattern);

        pattern = Pattern.compile(prePattern);
        matcher = pattern.matcher(lineBuf);

        out.println(HEADER_XML);
        printElement("tissue_expression",0);
        out.println("\t<source_id>" + CDATA_START +
convertedQuery + CDATA_END + "</source_id>");

        while (matcher.find()) {           // results are complete

            //System.out.println("This part works");

            pattern = Pattern.compile(mainPattern);
            matcher = pattern.matcher(lineBuf);

            if(matcher.find()) {

                //out.println("\t<source_id>" +
CDATA_START + convertedQuery + CDATA_END + "</source_id>");
                printElement("chrom", matcher.group(1),1);

```

```

matcher.group(2),1);
matcher.group(3),1);

printElement("chromStart",
printElement("chromEnd",
printElement("name", matcher.group(4),1);
printElement("score", matcher.group(5),1);
printElement("strand", matcher.group(6),1);
printElement("reserved", matcher.group(7),1);
String chromStart = matcher.group(8);
chromStart = chromStart.replaceAll("\\\\", "\\ ");
printElement("chromStarts", chromStart,1);
printElement("expCount", matcher.group(9),1);
/*
int i;
//System.out.println(matcher.groupCount());
for(i=10; i<matcher.groupCount(); i++) {
    printElement("expId", 2);
    printElement("Source", 3);
    printElement("ID", new
Integer(i).toString(), 4);

    printElement("DB", "database", 4);
    printElement("/Source", 3);
    printElement("Value",
matcher.group(i), 3);

    printElement("/expId", 2);
}
*/
printElement("expId", 1);
//printElement("Source", 2);
printElement("ID", new Integer(0).toString(),
2);

printElement("Name", "fetal brain", 2);
//printElement("/Source", 2);
printElement("Value", matcher.group(10), 2);
printElement("/expId", 1);

printElement("expId", 1);
printElement("ID", new Integer(1).toString(),
2);

printElement("Name", "whole brain", 2);
printElement("Value", matcher.group(11), 2);

```

```

printElement("/expId", 1);

printElement("expId", 1);
printElement("ID", new Integer(2).toString(),
2);

printElement("Name", "temporal lobe", 2);
printElement("Value", matcher.group(12), 2);
printElement("/expId", 1);

printElement("expId", 1);
printElement("ID", new Integer(3).toString(),
2);

printElement("Name", "parietal lobe", 2);
printElement("Value", matcher.group(13), 2);
printElement("/expId", 1);

printElement("expId", 1);
printElement("ID", new Integer(4).toString(),
2);

printElement("Name", "occipital lobe", 2);
printElement("Value", matcher.group(14), 2);
printElement("/expId", 1);

printElement("expId", 1);
printElement("ID", new Integer(5).toString(),
2);

printElement("Name", "prefrontal cortex", 2);
printElement("Value", matcher.group(15), 2);
printElement("/expId", 1);

printElement("expId", 1);
printElement("ID", new Integer(6).toString(),
2);

printElement("Name", "cingulate cortex", 2);
printElement("Value", matcher.group(16), 2);
printElement("/expId", 1);

printElement("expId", 1);
printElement("ID", new Integer(7).toString(),
2);

printElement("Name", "cerebellum", 2);

```

```

printElement("Value", matcher.group(17), 2);
printElement("/expId", 1);

printElement("expId", 1);
printElement("ID", new Integer(8).toString(),
2);

printElement("Name", "cerebellum peduncles",
2);

printElement("Value", matcher.group(18), 2);
printElement("/expId", 1);

printElement("expId", 1);
printElement("ID", new Integer(9).toString(),
2);

printElement("Name", "amygdala", 2);
printElement("Value", matcher.group(19), 2);
printElement("/expId", 1);

printElement("expId", 1);
printElement("ID", new Integer(10).toString(),
2);

printElement("Name", "hypothalamus", 2);
printElement("Value", matcher.group(20), 2);
printElement("/expId", 1);

printElement("expId", 1);
printElement("ID", new Integer(11).toString(),
2);

printElement("Name", "thalamus", 2);
printElement("Value", matcher.group(21), 2);
printElement("/expId", 1);

printElement("expId", 1);
printElement("ID", new Integer(12).toString(),
2);

printElement("Name", "subthalamic nucleus",
2);

printElement("Value", matcher.group(22), 2);
printElement("/expId", 1);

printElement("expId", 1);

```

```
2);
    printElement("ID", new Integer(13).toString(),
    printElement("Name", "caudate nucleus", 2);
    printElement("Value", matcher.group(23), 2);
    printElement("/expId", 1);

    printElement("expId", 1);
    printElement("ID", new Integer(14).toString(),
2);
    printElement("Name", "globus pallidus", 2);
    printElement("Value", matcher.group(24), 2);
    printElement("/expId", 1);

    printElement("expId", 1);
    printElement("ID", new Integer(15).toString(),
2);
    printElement("Name", "olfactory bulb", 2);
    printElement("Value", matcher.group(25), 2);
    printElement("/expId", 1);

    printElement("expId", 1);
    printElement("ID", new Integer(16).toString(),
2);
    printElement("Name", "pons", 2);
    printElement("Value", matcher.group(26), 2);
    printElement("/expId", 1);

    printElement("expId", 1);
    printElement("ID", new Integer(17).toString(),
2);
    printElement("Name", "medulla oblongata", 2);
    printElement("Value", matcher.group(27), 2);
    printElement("/expId", 1);

    printElement("expId", 1);
    printElement("ID", new Integer(18).toString(),
2);
    printElement("Name", "spinal cord", 2);
    printElement("Value", matcher.group(28), 2);
    printElement("/expId", 1);
```

```

printElement("expId", 1);
printElement("ID", new Integer(19).toString(),
2);

printElement("Name", "ciliary ganglion", 2);
printElement("Value", matcher.group(29), 2);
printElement("/expId", 1);

printElement("expId", 1);
printElement("ID", new Integer(20).toString(),
2);

printElement("Name", "trigeminal ganglion",
2);

printElement("Value", matcher.group(30), 2);
printElement("/expId", 1);

printElement("expId", 1);
printElement("ID", new Integer(21).toString(),
2);

printElement("Name", "superior cervical
ganglion", 2);

printElement("Value", matcher.group(31), 2);
printElement("/expId", 1);

printElement("expId", 1);
printElement("ID", new Integer(22).toString(),
2);

printElement("Name", "dorsal root ganglion",
2);

printElement("Value", matcher.group(32), 2);
printElement("/expId", 1);

printElement("expId", 1);
printElement("ID", new Integer(23).toString(),
2);

printElement("Name", "thymus", 2);
printElement("Value", matcher.group(33), 2);
printElement("/expId", 1);

printElement("expId", 1);
printElement("ID", new Integer(24).toString(),
2);

```

```

printElement("Name", "tonsil", 2);
printElement("Value", matcher.group(34), 2);
printElement("/expId", 1);

printElement("expId", 1);
printElement("ID", new Integer(25).toString(),
2);

printElement("Name", "lymph node", 2);
printElement("Value", matcher.group(35), 2);
printElement("/expId", 1);

printElement("expId", 1);
printElement("ID", new Integer(26).toString(),
2);

printElement("Name", "bone marrow", 2);
printElement("Value", matcher.group(36), 2);
printElement("/expId", 1);

printElement("expId", 1);
printElement("ID", new Integer(27).toString(),
2);

erythroid", 2);

printElement("Name", "BM-CD71+ early

printElement("Value", matcher.group(37), 2);
printElement("/expId", 1);

printElement("expId", 1);
printElement("ID", new Integer(28).toString(),
2);

printElement("Name", "BM-CD33+ myeloid",
2);

printElement("Value", matcher.group(38), 2);
printElement("/expId", 1);

printElement("expId", 1);
printElement("ID", new Integer(29).toString(),
2);

endothelial", 2);

printElement("Value", matcher.group(39), 2);
printElement("/expId", 1);

```

```

printElement("expId", 1);
printElement("ID", new Integer(30).toString(),
2);

printElement("Name", "BM-CD34+", 2);
printElement("Value", matcher.group(40), 2);
printElement("/expId", 1);

printElement("expId", 1);
printElement("ID", new Integer(31).toString(),
2);

printElement("Name", "whole blood", 2);
printElement("Value", matcher.group(41), 2);
printElement("/expId", 1);

printElement("expId", 1);
printElement("ID", new Integer(32).toString(),
2);

printElement("Name", "PB-BDCA4+ dendritic
cells", 2);

printElement("Value", matcher.group(42), 2);
printElement("/expId", 1);

printElement("expId", 1);
printElement("ID", new Integer(33).toString(),
2);

printElement("Name", "PB-CD14+
monocytes", 2);

printElement("Value", matcher.group(43), 2);
printElement("/expId", 1);

printElement("expId", 1);
printElement("ID", new Integer(34).toString(),
2);

printElement("Name", "PB-CD56+ NKCells",
2);

printElement("Value", matcher.group(44), 2);
printElement("/expId", 1);

printElement("expId", 1);

```

```

printElement("ID", new Integer(35).toString(),
2);

printElement("Name", "PB-CD4+ Tcells", 2);
printElement("Value", matcher.group(45), 2);
printElement("/expId", 1);

printElement("expId", 1);
printElement("ID", new Integer(36).toString(),
2);

printElement("Name", "PB-CD8+ Tcells", 2);
printElement("Value", matcher.group(46), 2);
printElement("/expId", 1);

printElement("expId", 1);
printElement("ID", new Integer(37).toString(),
2);

printElement("Name", "PB-CD19+ Bcells", 2);
printElement("Value", matcher.group(47), 2);
printElement("/expId", 1);

printElement("expId", 1);
printElement("ID", new Integer(38).toString(),
2);

printElement("Name", "leukemia
lymphoblastic(molt4)", 2);

printElement("Value", matcher.group(48), 2);
printElement("/expId", 1);

printElement("expId", 1);
printElement("ID", new Integer(39).toString(),
2);

printElement("Name", "721 B lymphoblasts",
2);

printElement("Value", matcher.group(49), 2);
printElement("/expId", 1);

printElement("expId", 1);
printElement("ID", new Integer(40).toString(),
2);

printElement("Name", "lymphoma Burkitts
Raji", 2);

```

```

printElement("Value", matcher.group(50), 2);
printElement("/expId", 1);

printElement("expId", 1);
printElement("ID", new Integer(41).toString(),
2);

printElement("Name", "leukemia
promyelocytic(hl60)", 2);

printElement("Value", matcher.group(51), 2);
printElement("/expId", 1);

printElement("expId", 1);
printElement("ID", new Integer(42).toString(),
2);

printElement("Name", "lymphoma Burkitts
Daudi", 2);

printElement("Value", matcher.group(52), 2);
printElement("/expId", 1);

printElement("expId", 1);
printElement("ID", new Integer(43).toString(),
2);

printElement("Name", "leukemia chronic
myelogenous(k562)", 2);

printElement("Value", matcher.group(53), 2);
printElement("/expId", 1);

printElement("expId", 1);
printElement("ID", new Integer(44).toString(),
2);

printElement("Name", "colorectal
adenocarcinoma", 2);

printElement("Value", matcher.group(54), 2);
printElement("/expId", 1);

printElement("expId", 1);
printElement("ID", new Integer(45).toString(),
2);

printElement("Name", "appendix", 2);
printElement("Value", matcher.group(55), 2);
printElement("/expId", 1);

```

```

printElement("expId", 1);
printElement("ID", new Integer(46).toString(),
2);

printElement("Name", "skin", 2);
printElement("Value", matcher.group(56), 2);
printElement("/expId", 1);

printElement("expId", 1);
printElement("ID", new Integer(47).toString(),
2);

printElement("Name", "adipocyte", 2);
printElement("Value", matcher.group(57), 2);
printElement("/expId", 1);

printElement("expId", 1);
printElement("ID", new Integer(48).toString(),
2);

printElement("Name", "fetal thyroid", 2);
printElement("Value", matcher.group(58), 2);
printElement("/expId", 1);

printElement("expId", 1);
printElement("ID", new Integer(49).toString(),
2);

printElement("Name", "thyroid", 2);
printElement("Value", matcher.group(59), 2);
printElement("/expId", 1);

printElement("expId", 1);
printElement("ID", new Integer(50).toString(),
2);

printElement("Name", "pituitary gland", 2);
printElement("Value", matcher.group(60), 2);
printElement("/expId", 1);

printElement("expId", 1);
printElement("ID", new Integer(51).toString(),
2);

printElement("Name", "adrenal gland", 2);
printElement("Value", matcher.group(61), 2);

```

```
printElement("/expId", 1);

printElement("expId", 1);
printElement("ID", new Integer(52).toString(),
2);

printElement("Name", "adrenal cortex", 2);
printElement("Value", matcher.group(62), 2);
printElement("/expId", 1);

printElement("expId", 1);
printElement("ID", new Integer(53).toString(),
2);

printElement("Name", "prostate", 2);
printElement("Value", matcher.group(63), 2);
printElement("/expId", 1);

printElement("expId", 1);
printElement("ID", new Integer(54).toString(),
2);

printElement("Name", "salivary gland", 2);
printElement("Value", matcher.group(64), 2);
printElement("/expId", 1);

printElement("expId", 1);
printElement("ID", new Integer(55).toString(),
2);

printElement("Name", "pancreas", 2);
printElement("Value", matcher.group(65), 2);
printElement("/expId", 1);

printElement("expId", 1);
printElement("ID", new Integer(56).toString(),
2);

printElement("Name", "pancreatic islets", 2);
printElement("Value", matcher.group(66), 2);
printElement("/expId", 1);

printElement("expId", 1);
printElement("ID", new Integer(57).toString(),
2);
```

```
printElement("Name", "atrioventricular node",
2);
printElement("Value", matcher.group(67), 2);
printElement("/expId", 1);

printElement("expId", 1);
printElement("ID", new Integer(58).toString(),
2);

printElement("Name", "heart", 2);
printElement("Value", matcher.group(68), 2);
printElement("/expId", 1);

printElement("expId", 1);
printElement("ID", new Integer(59).toString(),
2);

printElement("Name", "cardiac myocytes", 2);
printElement("Value", matcher.group(69), 2);
printElement("/expId", 1);

printElement("expId", 1);
printElement("ID", new Integer(60).toString(),
2);

printElement("Name", "skeletal muscle", 2);
printElement("Value", matcher.group(70), 2);
printElement("/expId", 1);

printElement("expId", 1);
printElement("ID", new Integer(61).toString(),
2);

printElement("Name", "tongue", 2);
printElement("Value", matcher.group(71), 2);
printElement("/expId", 1);

printElement("expId", 1);
printElement("ID", new Integer(62).toString(),
2);

printElement("Name", "smooth muscle", 2);
printElement("Value", matcher.group(72), 2);
printElement("/expId", 1);

printElement("expId", 1);
```

```

2);
    printElement("ID", new Integer(63).toString(),
    printElement("Name", "uterus", 2);
    printElement("Value", matcher.group(73), 2);
    printElement("/expId", 1);

    printElement("expId", 1);
    printElement("ID", new Integer(64).toString(),
2);

    printElement("Name", "uterus corpus", 2);
    printElement("Value", matcher.group(74), 2);
    printElement("/expId", 1);

    printElement("expId", 1);
    printElement("ID", new Integer(65).toString(),
2);

    printElement("Name", "trachea", 2);
    printElement("Value", matcher.group(75), 2);
    printElement("/expId", 1);

    printElement("expId", 1);
    printElement("ID", new Integer(66).toString(),
2);

    printElement("Name", "bronchial epithelial
cells", 2);

    printElement("Value", matcher.group(76), 2);
    printElement("/expId", 1);

    printElement("expId", 1);
    printElement("ID", new Integer(67).toString(),
2);

    printElement("Name", "fetal lung", 2);
    printElement("Value", matcher.group(77), 2);
    printElement("/expId", 1);

    printElement("expId", 1);
    printElement("ID", new Integer(68).toString(),
2);

    printElement("Name", "lung", 2);
    printElement("Value", matcher.group(78), 2);
    printElement("/expId", 1);

```

```

printElement("expId", 1);
printElement("ID", new Integer(69).toString(),
2);

printElement("Name", "kidney", 2);
printElement("Value", matcher.group(79), 2);
printElement("/expId", 1);

printElement("expId", 1);
printElement("ID", new Integer(70).toString(),
2);

printElement("Name", "fetal liver", 2);
printElement("Value", matcher.group(80), 2);
printElement("/expId", 1);

printElement("expId", 1);
printElement("ID", new Integer(71).toString(),
2);

printElement("Name", "liver", 2);
printElement("Value", matcher.group(81), 2);
printElement("/expId", 1);

printElement("expId", 1);
printElement("ID", new Integer(72).toString(),
2);

printElement("Name", "placenta", 2);
printElement("Value", matcher.group(82), 2);
printElement("/expId", 1);

printElement("expId", 1);
printElement("ID", new Integer(73).toString(),
2);

printElement("Name", "testis", 2);
printElement("Value", matcher.group(83), 2);
printElement("/expId", 1);

printElement("expId", 1);
printElement("ID", new Integer(74).toString(),
2);

printElement("Name", "testis Leydig cell", 2);
printElement("Value", matcher.group(84), 2);

```

```

printElement("/expId", 1);

printElement("expId", 1);
printElement("ID", new Integer(75).toString(),
2);

printElement("Name", "testis germ cell", 2);
printElement("Value", matcher.group(85), 2);
printElement("/expId", 1);

printElement("expId", 1);
printElement("ID", new Integer(76).toString(),
2);

printElement("Name", "testis interstitial", 2);
printElement("Value", matcher.group(86), 2);
printElement("/expId", 1);

printElement("expId", 1);
printElement("ID", new Integer(77).toString(),
2);

printElement("Name", "testis seminiferous
tubule", 2);

printElement("Value", matcher.group(87), 2);
printElement("/expId", 1);

printElement("expId", 1);
printElement("ID", new Integer(78).toString(),
2);

printElement("Name", "ovary", 2);
printElement("Value", matcher.group(88), 2);
printElement("/expId", 1);

printElement("tissue", matcher.group(89),1);
}
}

printElement("/tissue_expression",0);

br.close();

```

```

    } //end of try

    catch (Exception e) {

        out.println(HEADER_XML);
        System.err.println(e.getClass() + ": " + e.getMessage());
        e.printStackTrace();

    } //end of catch

} //end of processRequest

public String constructQuery(Iterator queryIterator) {

    StringBuffer termBuf = new StringBuffer("");
    HashMap groupMap = new HashMap();
    /*
    String ttype = null;
    String tvalue = null;
    String field = null;
    String hexencoded = null;
    String retQuery = null;
    String db = null;
    String dbvalue = null;
    */

    while (queryIterator.hasNext()) {
        // Retrieve the term values that were parsed from the query
        URL
            HashMap queryMap = (HashMap) queryIterator.next();
            ttype = (String)
                queryMap.get(AttribOption.OPTION_PARAM_NAME);
            tvalue = (String)
                queryMap.get(UCSCTISSUEEXP_OPTION_MAP_TVALUE);
            field = (String)
                queryMap.get(UCSCTISSUEEXP_OPTION_MAP_FIELD);
            db = (String)
                queryMap.get(UCSCTISSUEEXP_OPTION_MAP_DB);
    }
}

```

```

        dbvalue = (String)
queryMap.get(UCSCTISSUEEXP_OPTION_MAP_DBVALUE);

        hexencoded = (String)
queryMap.get(UCSCTISSUEEXP_OPTION_MAP_HEXENCODED);

        // Convert term values from hex to ascii if necessary.
        if (hexencoded != null && hexencoded.equals("true")) {
            tvalue = StringUtility.convertHexToASCII(tvalue);
            //dbvalue =
StringUtility.convertHexToASCII(dbvalue);
        }

        //retQuery = field + "=" + tvalue + "&" + db + "=" + dbvalue;

    }

    //System.out.println ("This is the returned query: " + retQuery);
    //return retQuery;
    return tvalue;
} //end of constructQuery Method

public HashMap buildQueryMap (String convertedQuery) {
    //System.out.println(" decoded converted query: "+ convertedQuery);
    HashMap queryMap = new HashMap();
    StringTokenizer st = new StringTokenizer(convertedQuery, "&");
    while (st.hasMoreTokens())
    {
        StringTokenizer st1 = new StringTokenizer(st.nextToken(),
"=");

        String key = st1.nextToken();
        String value = (st1.hasMoreTokens()?st1.nextToken():"");
        //System.out.println(" key: "+key+", value: "+value);
        queryMap.put(key, new String[]{value});
        //System.out.println(" key: "+key+", value: "+value);
    }
    return queryMap;
}

```

```

/**
 * getUsage
 * @return
 */
public String getUsage() {
    return(
        HEADER_XML + "\n" +
        "<" + TAG_WRAPPER_SOURCE + ">\n" +
        " <help>\n" +
        " <usage>\n" +
        " <![CDATA[ \n" + HELP_TEXT +
        " ]]>\n" +
        " </usage>\n" +
        " </help>\n" +
        "</" + TAG_WRAPPER_SOURCE + ">"
    );
}

public void printElement(String tag, String content, int tabs) {
    String tabStr = "";
    content = content.trim();
    for(int o = 0; o < tabs; o++)
        tabStr += "\t";
    out.println(tabStr + "<" + tag + ">" + CDATA_START + content +
        CDATA_END + "</" + tag + ">");
}

public void printElement(String tag, int tabs) {
    String tabStr = "";
    for(int o = 0; o < tabs; o++)
        tabStr += "\t";
    out.println(tabStr + "<" + tag + ">");
}

public void terminate() {
    // TODO Auto-generated method stub
}
}

```


Appendix B: Protégé Data Model

```

; Thu Feb 12 20:59:27 PST 2009
;
;+ (version "3.3.1")
;+ (build "Build 430")

([BROWSER_SLOT_NAMES] of Property_List

  (properties
    [SNP_ProjectKB_Instance_20088]
    [SNP_ProjectKB_Instance_20089]
    [SNP_ProjectKB_Instance_20090]
    [SNP_ProjectKB_Instance_20091]
    [SNP_ProjectKB_Instance_20092]))

([CLSES_TAB] of Widget

  (is_hidden FALSE)
  (label "Classes")
  (property_list [ProjectKB_0212_00005])
  (widget_class_name
"edu.stanford.smi.protege.widget.ClsesTab"))

([FORMS_TAB] of Widget

  (is_hidden FALSE)
  (label "Forms")
  (property_list [ProjectKB_0212_00034])
  (widget_class_name
"edu.stanford.smi.protege.widget.FormsTab"))

([GeneseekDBofDBs_ProjectKB_00009] of Widget

  (height 120)
  (is_hidden FALSE)
  (name "Name")
  (property_list [GeneseekDBofDBs_ProjectKB_00109])
  (widget_class_name
"edu.stanford.smi.protege.widget.StringListWidget")
  (width 250)
  (x 0)
  (y 0))

([GeneseekDBofDBs_ProjectKB_00020] of Widget

  (height 180)
  (is_hidden FALSE)

```

```

      (name "DBContains")
      (property_list [GeneseekDBofDBs_ProjectKB_00021])
      (widget_class_name
"edu.stanford.smi.protege.widget.InstanceListWidget")
      (width 250)
      (x 260)
      (y 0))

([GeneseekDBofDBs_ProjectKB_00021] of Property_List

  (properties
    [GeneseekDBofDBs_ProjectKB_00022]
    [GeneseekDBofDBs_ProjectKB_00023]
    [GeneseekDBofDBs_ProjectKB_00024]
    [GeneseekDBofDBs_ProjectKB_00025]
    [GeneseekDBofDBs_ProjectKB_00026]
    [GeneseekDBofDBs_ProjectKB_00027]
    [KolkerMediatedSchema_ProjectKB_Instance_107]
    [KolkerMediatedSchema_ProjectKB_Instance_108]
    [SNPTest_ProjectKB_Instance_20154]))

([GeneseekDBofDBs_ProjectKB_00022] of Boolean

  (boolean_value TRUE)
  (name "ButtonDisplayed-View Selected Instances"))

([GeneseekDBofDBs_ProjectKB_00023] of Boolean

  (boolean_value TRUE)
  (name "ButtonDisplayed-Create Instance"))

([GeneseekDBofDBs_ProjectKB_00024] of Boolean

  (boolean_value FALSE)
  (name "ButtonDisplayed-References"))

([GeneseekDBofDBs_ProjectKB_00025] of Boolean

  (boolean_value TRUE)
  (name "ButtonDisplayed-Add"))

([GeneseekDBofDBs_ProjectKB_00026] of Boolean

  (boolean_value TRUE)
  (name "ButtonDisplayed-Remove"))

([GeneseekDBofDBs_ProjectKB_00027] of Boolean

  (boolean_value FALSE)
  (name "ButtonDisplayed-Delete Selected Instances"))

```

```
([GeneseekDBofDBs_ProjectKB_00041] of Integer
  (integer_value 507)
  (name "ClsesTab.left_right"))

([GeneseekDBofDBs_ProjectKB_00042] of Integer
  (integer_value 1002)
  (name "ClsesTab.left.top_bottom"))

([GeneseekDBofDBs_ProjectKB_00043] of Integer
  (integer_value 254)
  (name "SlotsTab.left_right"))

([GeneseekDBofDBs_ProjectKB_00044] of Integer
  (integer_value 350)
  (name "SlotTab.left.top_bottom"))

([GeneseekDBofDBs_ProjectKB_00045] of Integer
  (integer_value 240)
  (name "FormsTab.left_right"))

([GeneseekDBofDBs_ProjectKB_00046] of Integer
  (integer_value 381)
  (name "InstancesTab.left_right"))

([GeneseekDBofDBs_ProjectKB_00047] of Integer
  (integer_value 530)
  (name "InstancesTab.right.left_right"))

([GeneseekDBofDBs_ProjectKB_00048] of String
  (name "SearchTab_Query"))

([GeneseekDBofDBs_ProjectKB_00109] of Property_List
  (properties
    [GeneseekDBofDBs_ProjectKB_00110]
    [GeneseekDBofDBs_ProjectKB_00111]
    [GeneseekDBofDBs_ProjectKB_00112]))

([GeneseekDBofDBs_ProjectKB_00110] of Boolean
  (boolean_value TRUE)
  (name "ButtonDisplayed-Edit the selected strings"))
```

```

([GeneseekDBofDBs_ProjectKB_00111] of Boolean
  (boolean_value TRUE)
  (name "ButtonDisplayed-Create a new string"))

([GeneseekDBofDBs_ProjectKB_00112] of Boolean
  (boolean_value TRUE)
  (name "ButtonDisplayed-Remove"))

([GeneseekDBofDBs_ProjectKB_00113] of Widget
  (height 60)
  (is_hidden FALSE)
  (name "Source")
  (property_list [GeneseekDBofDBs_ProjectKB_00114])
  (widget_class_name
"edu.stanford.smi.protege.widget.InstanceFieldWidget")
  (width 250)
  (x 0)
  (y 120))

([GeneseekDBofDBs_ProjectKB_00114] of Property_List
  (properties
    [GeneseekDBofDBs_ProjectKB_00115]
    [GeneseekDBofDBs_ProjectKB_00116]
    [GeneseekDBofDBs_ProjectKB_00117]
    [GeneseekDBofDBs_ProjectKB_00118]
    [GeneseekDBofDBs_ProjectKB_00119]
    [GeneseekDBofDBs_ProjectKB_00120]))

([GeneseekDBofDBs_ProjectKB_00115] of Boolean
  (boolean_value TRUE)
  (name "ButtonDisplayed-View Instance"))

([GeneseekDBofDBs_ProjectKB_00116] of Boolean
  (boolean_value FALSE)
  (name "ButtonDisplayed-References"))

([GeneseekDBofDBs_ProjectKB_00117] of Boolean
  (boolean_value TRUE)
  (name "ButtonDisplayed-Create Instance"))

([GeneseekDBofDBs_ProjectKB_00118] of Boolean
  (boolean_value TRUE)
  (name "ButtonDisplayed-Add"))

```

```

([GeneseekDBofDBs_ProjectKB_00119] of Boolean
  (boolean_value TRUE)
  (name "ButtonDisplayed-Remove"))

([GeneseekDBofDBs_ProjectKB_00120] of Boolean
  (boolean_value FALSE)
  (name "ButtonDisplayed-Delete Instance"))

([GeneseekDBofDBs_ProjectKB_00204] of Widget
  (is_hidden FALSE)
  (name ":STANDARD-FACET")
  (property_list [KB_862621_Instance_29])
  (widget_class_name
"edu.stanford.smi.protege.widget.FormWidget"))

([GeneseekDBofDBs_ProjectKB_00209] of Widget
  (is_hidden FALSE)
  (name ":INSTANCE-ANNOTATION")
  (property_list [KB_862621_Instance_33])
  (widget_class_name
"edu.stanford.smi.protege.widget.FormWidget"))

([GeneseekDBofDBs_ProjectKB_00218] of String
  (name "horizontal_stretcher"))

([GeneseekDBofDBs_ProjectKB_00219] of String
  (name "vertical_stretcher"))

([GeneseekDBofDBs_ProjectKB_00256] of Widget
  (height 1010)
  (is_hidden FALSE)
  (name "Relationships")
  (property_list [GeneseekDBofDBs_ProjectKB_00743])
  (widget_class_name
"edu.stanford.smi.protege.widget.FormWidget")
  (width 1320)
  (x 0)
  (y 0))

([GeneseekDBofDBs_ProjectKB_00302] of Widget
  (height 606)
  (is_hidden FALSE)

```

```

    (name "Mapping")
    (property_list [GeneseekDBofDBs_ProjectKB_00303])
    (widget_class_name
"edu.stanford.smi.protege.widget.FormWidget")
    (width 756)
    (x 0)
    (y 0))

```

```

([GeneseekDBofDBs_ProjectKB_00303] of Property_List

```

```

    (properties
      [GeneseekDBofDBs_ProjectKB_01079]
      [GeneseekDBofDBs_ProjectKB_01085]
      [GeneseekDBofDBs_ProjectKB_01087]
      [GeneseekDBofDBs_ProjectKB_00633]))

```

```

([GeneseekDBofDBs_ProjectKB_00361] of Widget

```

```

    (height 606)
    (is_hidden FALSE)
    (name "Databases")
    (property_list [GeneseekDBofDBs_ProjectKB_00362])
    (widget_class_name
"edu.stanford.smi.protege.widget.FormWidget")
    (width 756)
    (x 0)
    (y 0))

```

```

([GeneseekDBofDBs_ProjectKB_00362] of Property_List

```

```

    (properties
      [GeneseekDBofDBs_ProjectKB_00368]
      [GeneseekDBofDBs_ProjectKB_00375]
      [GeneseekDBofDBs_ProjectKB_00020]
      [GeneseekDBofDBs_ProjectKB_00870]
      [GeneseekDBofDBs_ProjectKB_00432]
      [GeneseekDBofDBs_ProjectKB_01040]))

```

```

([GeneseekDBofDBs_ProjectKB_00368] of Widget

```

```

    (height 60)
    (is_hidden FALSE)
    (name "DBName")
    (property_list [GeneseekDBofDBs_ProjectKB_00369])
    (widget_class_name
"edu.stanford.smi.protege.widget.TextFieldWidget")
    (width 250)
    (x 0)
    (y 0))

```

```

([GeneseekDBofDBs_ProjectKB_00369] of Property_List

```

```

)

([GeneseekDBofDBs_ProjectKB_00375] of Property_List
  (name "layout properties"))

([GeneseekDBofDBs_ProjectKB_00432] of Widget
  (height 120)
  (is_hidden FALSE)
  (name "DBMapping")
  (property_list [GeneseekDBofDBs_ProjectKB_00433])
  (widget_class_name
"edu.stanford.smi.protege.widget.StringListWidget")
  (width 250)
  (x 0)
  (y 180))

([GeneseekDBofDBs_ProjectKB_00433] of Property_List
  (properties
    [GeneseekDBofDBs_ProjectKB_00434]
    [GeneseekDBofDBs_ProjectKB_00435]
    [GeneseekDBofDBs_ProjectKB_00436]))

([GeneseekDBofDBs_ProjectKB_00434] of Boolean
  (boolean_value TRUE)
  (name "ButtonDisplayed-Edit the selected strings"))

([GeneseekDBofDBs_ProjectKB_00435] of Boolean
  (boolean_value TRUE)
  (name "ButtonDisplayed-Create a new string"))

([GeneseekDBofDBs_ProjectKB_00436] of Boolean
  (boolean_value TRUE)
  (name "ButtonDisplayed-Remove"))

([GeneseekDBofDBs_ProjectKB_00596] of Widget
  (height 1010)
  (is_hidden FALSE)
  (name "EntityMapping")
  (property_list [GeneseekDBofDBs_ProjectKB_01184])
  (widget_class_name
"edu.stanford.smi.protege.widget.FormWidget")
  (width 1320)
  (x 0)
  (y 0))

```

```

([GeneseekDBofDBs_ProjectKB_00633] of Widget

    (height 120)
    (is_hidden FALSE)
    (name "Notes")
    (property_list [GeneseekDBofDBs_ProjectKB_00634])
    (widget_class_name
"edu.stanford.smi.protege.widget.TextAreaWidget")
    (width 250)
    (x 0)
    (y 180))

([GeneseekDBofDBs_ProjectKB_00634] of Property_List
)

([GeneseekDBofDBs_ProjectKB_00640] of Map

    (entries [SNP_ProjectKB_Instance_20098])
    (referenced_maps [SNP_ProjectKB_Instance_20093]))

([GeneseekDBofDBs_ProjectKB_00679] of Widget

    (height 605)
    (is_hidden FALSE)
    (name "Entity")
    (property_list [GeneseekDBofDBs_ProjectKB_00740])
    (widget_class_name
"edu.stanford.smi.protege.widget.FormWidget")
    (width 746)
    (x 0)
    (y 0))

([GeneseekDBofDBs_ProjectKB_00740] of Property_List

    (name "class widget properties")
    (properties
        [GeneseekDBofDBs_ProjectKB_00764]
        [GeneseekDBofDBs_ProjectKB_00009]
        [GeneseekDBofDBs_ProjectKB_00113]))

([GeneseekDBofDBs_ProjectKB_00743] of Property_List

    (properties [GeneseekDBofDBs_ProjectKB_00744]))

([GeneseekDBofDBs_ProjectKB_00744] of Property_List

    (name "layout properties")
    (properties
        [GeneseekDBofDBs_ProjectKB_00745]
        [GeneseekDBofDBs_ProjectKB_00746]))

```

```
([GeneseekDBofDBs_ProjectKB_00745] of String
  (name "horizontal_stretcher"))

([GeneseekDBofDBs_ProjectKB_00746] of String
  (name "vertical_stretcher"))

([GeneseekDBofDBs_ProjectKB_00764] of Property_List
  (name "layout properties")
  (properties
    [GeneseekDBofDBs_ProjectKB_00218]
    [GeneseekDBofDBs_ProjectKB_00219]))

([GeneseekDBofDBs_ProjectKB_00870] of Widget
  (height 120)
  (is_hidden FALSE)
  (name "DBPrimaryEntity")
  (property_list [GeneseekDBofDBs_ProjectKB_00871])
  (widget_class_name
"edu.stanford.smi.protege.widget.ClsListWidget")
  (width 250)
  (x 0)
  (y 60))

([GeneseekDBofDBs_ProjectKB_00871] of Property_List
  (properties
    [GeneseekDBofDBs_ProjectKB_00872]
    [GeneseekDBofDBs_ProjectKB_00873]
    [GeneseekDBofDBs_ProjectKB_00874]))

([GeneseekDBofDBs_ProjectKB_00872] of Boolean
  (boolean_value TRUE)
  (name "ButtonDisplayed-View Selected Classes"))

([GeneseekDBofDBs_ProjectKB_00873] of Boolean
  (boolean_value TRUE)
  (name "ButtonDisplayed-Add"))

([GeneseekDBofDBs_ProjectKB_00874] of Boolean
  (boolean_value TRUE)
  (name "ButtonDisplayed-Remove"))

([GeneseekDBofDBs_ProjectKB_01040] of Widget
```

```

    (height 120)
    (is_hidden FALSE)
    (name "Notes")
    (property_list [GeneseekDBofDBs_ProjectKB_01041])
    (widget_class_name
"edu.stanford.smi.protege.widget.TextAreaWidget")
    (width 250)
    (x 0)
    (y 360))

([GeneseekDBofDBs_ProjectKB_01041] of Property_List
)

([GeneseekDBofDBs_ProjectKB_01079] of Property_List
    (name "layout properties"))

([GeneseekDBofDBs_ProjectKB_01085] of Widget
    (height 60)
    (is_hidden FALSE)
    (name "MappingName")
    (property_list [GeneseekDBofDBs_ProjectKB_01086])
    (widget_class_name
"edu.stanford.smi.protege.widget.TextFieldWidget")
    (width 250)
    (x 0)
    (y 0))

([GeneseekDBofDBs_ProjectKB_01086] of Property_List
)

([GeneseekDBofDBs_ProjectKB_01087] of Widget
    (height 120)
    (is_hidden FALSE)
    (name "DBMapping")
    (property_list [GeneseekDBofDBs_ProjectKB_01088])
    (widget_class_name
"edu.stanford.smi.protege.widget.StringListWidget")
    (width 250)
    (x 0)
    (y 60))

([GeneseekDBofDBs_ProjectKB_01088] of Property_List
    (properties
      [GeneseekDBofDBs_ProjectKB_01089]
      [GeneseekDBofDBs_ProjectKB_01090]
      [GeneseekDBofDBs_ProjectKB_01091]))

```

```

([GeneseekDBofDBs_ProjectKB_01089] of Boolean
  (boolean_value TRUE)
  (name "ButtonDisplayed-Edit the selected strings"))

([GeneseekDBofDBs_ProjectKB_01090] of Boolean
  (boolean_value TRUE)
  (name "ButtonDisplayed-Create a new string"))

([GeneseekDBofDBs_ProjectKB_01091] of Boolean
  (boolean_value TRUE)
  (name "ButtonDisplayed-Remove"))

([GeneseekDBofDBs_ProjectKB_01184] of Property_List
  (properties
    [GeneseekDBofDBs_ProjectKB_01186]
    [GeneseekDBofDBs_ProjectKB_01286]
    [GeneseekDBofDBs_ProjectKB_01389]
    [GeneseekDBofDBs_ProjectKB_01493]
    [GeneseekDBofDBs_ProjectKB_01495]
    [KolkerTest_ProjectKB_Instance_20039]))

([GeneseekDBofDBs_ProjectKB_01185] of Widget
  (height 605)
  (is_hidden FALSE)
  (name "EdgeMetaClass")
  (property_list [GeneseekDBofDBs_ProjectKB_01255])
  (widget_class_name
"edu.stanford.smi.protege.widget.FormWidget")
  (width 746)
  (x 0)
  (y 0))

([GeneseekDBofDBs_ProjectKB_01186] of Property_List
  (name "layout properties"))

([GeneseekDBofDBs_ProjectKB_01255] of Property_List
  (name "class widget properties")
  (properties
    [GeneseekDBofDBs_ProjectKB_01256]
    [GeneseekDBofDBs_ProjectKB_01264]
    [GeneseekDBofDBs_ProjectKB_01265]
    [GeneseekDBofDBs_ProjectKB_01266]
    [GeneseekDBofDBs_ProjectKB_01267]
  ))

```

```

[GeneseekDBofDBs_ProjectKB_01268]
[GeneseekDBofDBs_ProjectKB_01269]
[GeneseekDBofDBs_ProjectKB_01270]
[GeneseekDBofDBs_ProjectKB_01275]
[GeneseekDBofDBs_ProjectKB_01283]
[GeneseekDBofDBs_ProjectKB_01288]
[GeneseekDBofDBs_ProjectKB_01293]
[GeneseekDBofDBs_ProjectKB_01298]))

([GeneseekDBofDBs_ProjectKB_01256] of Widget

  (height 120)
  (label "Constraints")
  (name " :SLOT-CONSTRAINTS")
  (property_list [GeneseekDBofDBs_ProjectKB_01257])
  (widget_class_name
"edu.stanford.smi.protege.widget.ConstraintsWidget")
  (width 200)
  (x 400)
  (y 0))

([GeneseekDBofDBs_ProjectKB_01257] of Property_List

  (properties
    [GeneseekDBofDBs_ProjectKB_01258]
    [GeneseekDBofDBs_ProjectKB_01259]
    [GeneseekDBofDBs_ProjectKB_01260]
    [GeneseekDBofDBs_ProjectKB_01261]
    [GeneseekDBofDBs_ProjectKB_01262]
    [GeneseekDBofDBs_ProjectKB_01263]
    [KolkerMediatedSchema_ProjectKB_Instance_80]
    [KolkerMediatedSchema_ProjectKB_Instance_81]
    [SNPTest_ProjectKB_Instance_128]))

([GeneseekDBofDBs_ProjectKB_01258] of Boolean

  (boolean_value TRUE)
  (name "ButtonDisplayed-View Selected Instances"))

([GeneseekDBofDBs_ProjectKB_01259] of Boolean

  (boolean_value TRUE)
  (name "ButtonDisplayed-Create Instance"))

([GeneseekDBofDBs_ProjectKB_01260] of Boolean

  (boolean_value FALSE)
  (name "ButtonDisplayed-References"))

([GeneseekDBofDBs_ProjectKB_01261] of Boolean

```

```

        (boolean_value TRUE)
        (name "ButtonDisplayed-Add"))

    ([GeneseekDBofDBs_ProjectKB_01262] of Boolean

        (boolean_value TRUE)
        (name "ButtonDisplayed-Remove"))

    ([GeneseekDBofDBs_ProjectKB_01263] of Boolean

        (boolean_value FALSE)
        (name "ButtonDisplayed-Delete Selected Instances"))

    ([GeneseekDBofDBs_ProjectKB_01264] of Widget

        (name ":DIRECT-INSTANCES"))

    ([GeneseekDBofDBs_ProjectKB_01265] of Widget

        (name ":DIRECT-SUBCLASSES"))

    ([GeneseekDBofDBs_ProjectKB_01266] of Widget

        (name ":DIRECT-SUPERCLASSES"))

    ([GeneseekDBofDBs_ProjectKB_01267] of Widget

        (height 120)
        (label "Documentation")
        (name ":DOCUMENTATION")
        (property_list [SNPTest_ProjectKB_Instance_25])
        (widget_class_name
"edu.stanford.smi.protege.widget.DocumentationWidget")
        (width 200)
        (x 200)
        (y 0))

    ([GeneseekDBofDBs_ProjectKB_01268] of Widget

        (height 60)
        (label "Name")
        (name ":NAME")
        (property_list [SNPTest_ProjectKB_Instance_26])
        (widget_class_name
"edu.stanford.smi.protege.widget.InstanceNameWidget")
        (width 200)
        (x 0)
        (y 0))

    ([GeneseekDBofDBs_ProjectKB_01269] of Widget

```

```

    (height 60)
    (label "Role")
    (name ":ROLE")
    (property_list [SNPTest_ProjectKB_Instance_27])
    (widget_class_name
"edu.stanford.smi.protege.widget.RoleWidget")
    (width 200)
    (x 0)
    (y 60))

([GeneseekDBofDBs_ProjectKB_01270] of Widget

    (height 0)
    (name ":DIRECT-TYPE")
    (property_list [GeneseekDBofDBs_ProjectKB_01271])
    (width 0)
    (x 0)
    (y 0))

([GeneseekDBofDBs_ProjectKB_01271] of Property_List
)

([GeneseekDBofDBs_ProjectKB_01275] of Widget

    (height 300)
    (label "Template Slots")
    (name ":DIRECT-TEMPLATE-SLOTS")
    (property_list [GeneseekDBofDBs_ProjectKB_01276])
    (widget_class_name
"edu.stanford.smi.protege.widget.TemplateSlotsWidget")
    (width 600)
    (x 0)
    (y 180))

([GeneseekDBofDBs_ProjectKB_01276] of Property_List

    (properties
      [GeneseekDBofDBs_ProjectKB_01277]
      [GeneseekDBofDBs_ProjectKB_01278]
      [GeneseekDBofDBs_ProjectKB_01279]
      [GeneseekDBofDBs_ProjectKB_01280]
      [GeneseekDBofDBs_ProjectKB_01281]
      [GeneseekDBofDBs_ProjectKB_01282]
      [GeneseekDBofDBs_ProjectKB_01515]
      [GeneseekDBofDBs_ProjectKB_01516]))

([GeneseekDBofDBs_ProjectKB_01277] of Boolean

    (boolean_value TRUE)
    (name "ButtonDisplayed-View selected slots"))

```

```

([GeneseekDBofDBs_ProjectKB_01278] of Boolean
  (boolean_value TRUE)
  (name "ButtonDisplayed-View selected slots at class"))

([GeneseekDBofDBs_ProjectKB_01279] of Boolean
  (boolean_value TRUE)
  (name "ButtonDisplayed-Create slot and attach to class"))

([GeneseekDBofDBs_ProjectKB_01280] of Boolean
  (boolean_value TRUE)
  (name "ButtonDisplayed-Remove overrides from selected slots"))

([GeneseekDBofDBs_ProjectKB_01281] of Boolean
  (boolean_value TRUE)
  (name "ButtonDisplayed-Add"))

([GeneseekDBofDBs_ProjectKB_01282] of Boolean
  (boolean_value TRUE)
  (name "ButtonDisplayed-Remove"))

([GeneseekDBofDBs_ProjectKB_01283] of Property_List
  (name "layout properties")
  (properties
    [GeneseekDBofDBs_ProjectKB_01284]
    [GeneseekDBofDBs_ProjectKB_01285]))

([GeneseekDBofDBs_ProjectKB_01284] of String
  (name "horizontal_stretcher"))

([GeneseekDBofDBs_ProjectKB_01285] of String
  (name "vertical_stretcher"))

([GeneseekDBofDBs_ProjectKB_01286] of Widget
  (height 60)
  (is_hidden FALSE)
  (name "MappingName")
  (property_list [GeneseekDBofDBs_ProjectKB_01287])
  (widget_class_name
"edu.stanford.smi.protege.widget.TextFieldWidget")
  (width 250)
  (x 0)
  (y 0))

```

```

([GeneseekDBofDBs_ProjectKB_01287] of Property_List
)

([GeneseekDBofDBs_ProjectKB_01288] of Widget

    (height 60)
    (is_hidden FALSE)
    (name "HeadType")
    (property_list [GeneseekDBofDBs_ProjectKB_01289])
    (widget_class_name
"edu.stanford.smi.protege.widget.ClsFieldWidget")
    (width 200)
    (x 0)
    (y 120))

([GeneseekDBofDBs_ProjectKB_01289] of Property_List

    (properties
    [GeneseekDBofDBs_ProjectKB_01290]
    [GeneseekDBofDBs_ProjectKB_01291]
    [GeneseekDBofDBs_ProjectKB_01292]))

([GeneseekDBofDBs_ProjectKB_01290] of Boolean

    (boolean_value TRUE)
    (name "ButtonDisplayed-View Class"))

([GeneseekDBofDBs_ProjectKB_01291] of Boolean

    (boolean_value TRUE)
    (name "ButtonDisplayed-Add"))

([GeneseekDBofDBs_ProjectKB_01292] of Boolean

    (boolean_value TRUE)
    (name "ButtonDisplayed-Remove"))

([GeneseekDBofDBs_ProjectKB_01293] of Widget

    (height 60)
    (is_hidden FALSE)
    (name "TailType")
    (property_list [GeneseekDBofDBs_ProjectKB_01294])
    (widget_class_name
"edu.stanford.smi.protege.widget.ClsFieldWidget")
    (width 200)
    (x 200)
    (y 120))

([GeneseekDBofDBs_ProjectKB_01294] of Property_List

```

```

(properties
  [GeneseekDBofDBs_ProjectKB_01295]
  [GeneseekDBofDBs_ProjectKB_01296]
  [GeneseekDBofDBs_ProjectKB_01297]))

([GeneseekDBofDBs_ProjectKB_01295] of Boolean

  (boolean_value TRUE)
  (name "ButtonDisplayed-View Class"))

([GeneseekDBofDBs_ProjectKB_01296] of Boolean

  (boolean_value TRUE)
  (name "ButtonDisplayed-Add"))

([GeneseekDBofDBs_ProjectKB_01297] of Boolean

  (boolean_value TRUE)
  (name "ButtonDisplayed-Remove"))

([GeneseekDBofDBs_ProjectKB_01298] of Widget

  (height 60)
  (is_hidden FALSE)
  (name "InverseEdge")
  (property_list [GeneseekDBofDBs_ProjectKB_01299])
  (widget_class_name
"edu.stanford.smi.protege.widget.InstanceFieldWidget")
  (width 200)
  (x 400)
  (y 120))

([GeneseekDBofDBs_ProjectKB_01299] of Property_List

  (properties
    [GeneseekDBofDBs_ProjectKB_01300]
    [GeneseekDBofDBs_ProjectKB_01301]
    [GeneseekDBofDBs_ProjectKB_01302]
    [GeneseekDBofDBs_ProjectKB_01303]
    [GeneseekDBofDBs_ProjectKB_01304]
    [GeneseekDBofDBs_ProjectKB_01305]
    [KolkerMediatedSchema_ProjectKB_Instance_82]
    [SNPTest_ProjectKB_Instance_129]))

([GeneseekDBofDBs_ProjectKB_01300] of Boolean

  (boolean_value TRUE)
  (name "ButtonDisplayed-View Instance"))

([GeneseekDBofDBs_ProjectKB_01301] of Boolean

```

```

        (boolean_value FALSE)
        (name "ButtonDisplayed-References"))

([GeneseekDBofDBs_ProjectKB_01302] of Boolean

        (boolean_value TRUE)
        (name "ButtonDisplayed-Create Instance"))

([GeneseekDBofDBs_ProjectKB_01303] of Boolean

        (boolean_value TRUE)
        (name "ButtonDisplayed-Add"))

([GeneseekDBofDBs_ProjectKB_01304] of Boolean

        (boolean_value TRUE)
        (name "ButtonDisplayed-Remove"))

([GeneseekDBofDBs_ProjectKB_01305] of Boolean

        (boolean_value FALSE)
        (name "ButtonDisplayed-Delete Instance"))

([GeneseekDBofDBs_ProjectKB_01389] of Widget

        (height 120)
        (is_hidden FALSE)
        (name "DBMapping")
        (property_list [GeneseekDBofDBs_ProjectKB_01489])
        (widget_class_name
"edu.stanford.smi.protege.widget.StringListWidget")
        (width 250)
        (x 250)
        (y 60))

([GeneseekDBofDBs_ProjectKB_01489] of Property_List

        (properties
        [GeneseekDBofDBs_ProjectKB_01490]
        [GeneseekDBofDBs_ProjectKB_01491]
        [GeneseekDBofDBs_ProjectKB_01492]))

([GeneseekDBofDBs_ProjectKB_01490] of Boolean

        (boolean_value TRUE)
        (name "ButtonDisplayed-Edit the selected strings"))

([GeneseekDBofDBs_ProjectKB_01491] of Boolean

        (boolean_value TRUE)

```

```

        (name "ButtonDisplayed-Create a new string"))
    ([GeneseekDBofDBs_ProjectKB_01492] of Boolean
      (boolean_value TRUE)
      (name "ButtonDisplayed-Remove"))
    ([GeneseekDBofDBs_ProjectKB_01493] of Widget
      (height 120)
      (is_hidden FALSE)
      (name "Notes")
      (property_list [GeneseekDBofDBs_ProjectKB_01494])
      (widget_class_name
"edu.stanford.smi.protege.widget.TextAreaWidget")
      (width 250)
      (x 0)
      (y 60))
    ([GeneseekDBofDBs_ProjectKB_01494] of Property_List
    )
    ([GeneseekDBofDBs_ProjectKB_01495] of Widget
      (height 60)
      (is_hidden FALSE)
      (name "DBObject")
      (property_list [GeneseekDBofDBs_ProjectKB_01496])
      (widget_class_name
"edu.stanford.smi.protege.widget.ClsFieldWidget")
      (width 250)
      (x 250)
      (y 0))
    ([GeneseekDBofDBs_ProjectKB_01496] of Property_List
      (properties
        [GeneseekDBofDBs_ProjectKB_01497]
        [GeneseekDBofDBs_ProjectKB_01498]
        [GeneseekDBofDBs_ProjectKB_01499]))
    ([GeneseekDBofDBs_ProjectKB_01497] of Boolean
      (boolean_value TRUE)
      (name "ButtonDisplayed-View Class"))
    ([GeneseekDBofDBs_ProjectKB_01498] of Boolean
      (boolean_value TRUE)
      (name "ButtonDisplayed-Add"))

```

```

([GeneseekDBofDBs_ProjectKB_01499] of Boolean
  (boolean_value TRUE)
  (name "ButtonDisplayed-Remove"))

([GeneseekDBofDBs_ProjectKB_01500] of Property_List
  (properties
    [GeneseekDBofDBs_ProjectKB_01501]
    [GeneseekDBofDBs_ProjectKB_01502]
    [GeneseekDBofDBs_ProjectKB_01504]
    [GeneseekDBofDBs_ProjectKB_01509]
    [GeneseekDBofDBs_ProjectKB_01511]
    [GeneseekDBofDBs_ProjectKB_01513]
    [GeneseekDBofDBs_ProjectKB_01721]
    [GeneseekDBofDBs_ProjectKB_01729]
    [GeneseekDBofDBs_ProjectKB_01734]
    [GeneseekDBofDBs_ProjectKB_01742]
    [GeneseekDBofDBs_ProjectKB_01744]
    [GeneseekDBofDBs_ProjectKB_01752]
    [KolkerTest_ProjectKB_Instance_10012]
    [SNPTest_ProjectKB_Instance_10063]))

([GeneseekDBofDBs_ProjectKB_01501] of Property_List
  (name "layout properties"))

([GeneseekDBofDBs_ProjectKB_01502] of Widget
  (height 60)
  (is_hidden FALSE)
  (name "MappingName")
  (property_list [GeneseekDBofDBs_ProjectKB_01503])
  (widget_class_name
"edu.stanford.smi.protege.widget.TextFieldWidget")
  (width 250)
  (x 0)
  (y 0))

([GeneseekDBofDBs_ProjectKB_01503] of Property_List
)

([GeneseekDBofDBs_ProjectKB_01504] of Widget
  (height 120)
  (is_hidden FALSE)
  (name "DBMapping")
  (property_list [GeneseekDBofDBs_ProjectKB_01505])
  (widget_class_name
"edu.stanford.smi.protege.widget.StringListWidget")
  (width 250)

```

```

(x 250)
(y 180))

([GeneseekDBofDBs_ProjectKB_01505] of Property_List

  (properties
    [GeneseekDBofDBs_ProjectKB_01506]
    [GeneseekDBofDBs_ProjectKB_01507]
    [GeneseekDBofDBs_ProjectKB_01508]))

([GeneseekDBofDBs_ProjectKB_01506] of Boolean

  (boolean_value TRUE)
  (name "ButtonDisplayed-Edit the selected strings"))

([GeneseekDBofDBs_ProjectKB_01507] of Boolean

  (boolean_value TRUE)
  (name "ButtonDisplayed-Create a new string"))

([GeneseekDBofDBs_ProjectKB_01508] of Boolean

  (boolean_value TRUE)
  (name "ButtonDisplayed-Remove"))

([GeneseekDBofDBs_ProjectKB_01509] of Widget

  (height 120)
  (is_hidden FALSE)
  (name "Notes")
  (property_list [GeneseekDBofDBs_ProjectKB_01510])
  (widget_class_name
"edu.stanford.smi.protege.widget.TextAreaWidget")
  (width 250)
  (x 0)
  (y 60))

([GeneseekDBofDBs_ProjectKB_01510] of Property_List
)

([GeneseekDBofDBs_ProjectKB_01511] of Widget

  (height 60)
  (is_hidden FALSE)
  (name "__isContainment")
  (property_list [GeneseekDBofDBs_ProjectKB_01512])
  (widget_class_name
"edu.stanford.smi.protege.widget.CheckBoxWidget")
  (width 125)
  (x 125)
  (y 180))

```

```

([GeneseekDBofDBs_ProjectKB_01512] of Property_List
)

([GeneseekDBofDBs_ProjectKB_01513] of Widget

  (height 60)
  (is_hidden FALSE)
  (name "HeadDB")
  (property_list [GeneseekDBofDBs_ProjectKB_01514])
  (widget_class_name
"edu.stanford.smi.protege.widget.InstanceFieldWidget")
  (width 250)
  (x 0)
  (y 300))

([GeneseekDBofDBs_ProjectKB_01514] of Property_List

  (properties
    [GeneseekDBofDBs_ProjectKB_01616]
    [GeneseekDBofDBs_ProjectKB_01716]
    [GeneseekDBofDBs_ProjectKB_01717]
    [GeneseekDBofDBs_ProjectKB_01718]
    [GeneseekDBofDBs_ProjectKB_01719]
    [GeneseekDBofDBs_ProjectKB_01720]
    [KolkerMediatedSchema_ProjectKB_Instance_221]
    [SNPTest_ProjectKB_Instance_124]))

([GeneseekDBofDBs_ProjectKB_01515] of Boolean

  (boolean_value FALSE)
  (name "ButtonDisplayed-Move up"))

([GeneseekDBofDBs_ProjectKB_01516] of Boolean

  (boolean_value FALSE)
  (name "ButtonDisplayed-Move down"))

([GeneseekDBofDBs_ProjectKB_01616] of Boolean

  (boolean_value TRUE)
  (name "ButtonDisplayed-View Instance"))

([GeneseekDBofDBs_ProjectKB_01716] of Boolean

  (boolean_value FALSE)
  (name "ButtonDisplayed-References"))

([GeneseekDBofDBs_ProjectKB_01717] of Boolean

  (boolean_value TRUE)

```

```

        (name "ButtonDisplayed-Create Instance"))
    ([GeneseekDBofDBs_ProjectKB_01718] of Boolean
      (boolean_value TRUE)
      (name "ButtonDisplayed-Add"))
    ([GeneseekDBofDBs_ProjectKB_01719] of Boolean
      (boolean_value TRUE)
      (name "ButtonDisplayed-Remove"))
    ([GeneseekDBofDBs_ProjectKB_01720] of Boolean
      (boolean_value FALSE)
      (name "ButtonDisplayed-Delete Instance"))
    ([GeneseekDBofDBs_ProjectKB_01721] of Widget
      (height 60)
      (is_hidden FALSE)
      (name "TailDB")
      (property_list [GeneseekDBofDBs_ProjectKB_01722])
      (widget_class_name
"edu.stanford.smi.protege.widget.InstanceFieldWidget")
      (width 250)
      (x 0)
      (y 360))
    ([GeneseekDBofDBs_ProjectKB_01722] of Property_List
      (properties
        [GeneseekDBofDBs_ProjectKB_01723]
        [GeneseekDBofDBs_ProjectKB_01724]
        [GeneseekDBofDBs_ProjectKB_01725]
        [GeneseekDBofDBs_ProjectKB_01726]
        [GeneseekDBofDBs_ProjectKB_01727]
        [GeneseekDBofDBs_ProjectKB_01728]
        [KolkerMediatedSchema_ProjectKB_Instance_222]
        [SNPTTest_ProjectKB_Instance_125]))
    ([GeneseekDBofDBs_ProjectKB_01723] of Boolean
      (boolean_value TRUE)
      (name "ButtonDisplayed-View Instance"))
    ([GeneseekDBofDBs_ProjectKB_01724] of Boolean
      (boolean_value FALSE)
      (name "ButtonDisplayed-References"))

```

```

([GeneseekDBofDBs_ProjectKB_01725] of Boolean
  (boolean_value TRUE)
  (name "ButtonDisplayed-Create Instance"))

([GeneseekDBofDBs_ProjectKB_01726] of Boolean
  (boolean_value TRUE)
  (name "ButtonDisplayed-Add"))

([GeneseekDBofDBs_ProjectKB_01727] of Boolean
  (boolean_value TRUE)
  (name "ButtonDisplayed-Remove"))

([GeneseekDBofDBs_ProjectKB_01728] of Boolean
  (boolean_value FALSE)
  (name "ButtonDisplayed-Delete Instance"))

([GeneseekDBofDBs_ProjectKB_01729] of Widget
  (height 120)
  (is_hidden FALSE)
  (name "__creation")
  (property_list [GeneseekDBofDBs_ProjectKB_01730])
  (widget_class_name
"edu.stanford.smi.protege.widget.SymbolListWidget")
  (width 250)
  (x 250)
  (y 60))

([GeneseekDBofDBs_ProjectKB_01730] of Property_List
  (properties
    [GeneseekDBofDBs_ProjectKB_01731]
    [GeneseekDBofDBs_ProjectKB_01732]
    [GeneseekDBofDBs_ProjectKB_01733]))

([GeneseekDBofDBs_ProjectKB_01731] of Boolean
  (boolean_value TRUE)
  (name "ButtonDisplayed-Change the selected values"))

([GeneseekDBofDBs_ProjectKB_01732] of Boolean
  (boolean_value TRUE)
  (name "ButtonDisplayed-Create a new value"))

([GeneseekDBofDBs_ProjectKB_01733] of Boolean

```

```

    (boolean_value TRUE)
    (name "ButtonDisplayed-Remove"))

([GeneseekDBofDBs_ProjectKB_01734] of Widget

    (height 60)
    (is_hidden FALSE)
    (name "RelationshipType")
    (property_list [GeneseekDBofDBs_ProjectKB_01735])
    (widget_class_name
"edu.stanford.smi.protege.widget.InstanceFieldWidget")
    (width 250)
    (x 250)
    (y 0))

([GeneseekDBofDBs_ProjectKB_01735] of Property_List

    (properties
      [GeneseekDBofDBs_ProjectKB_01736]
      [GeneseekDBofDBs_ProjectKB_01737]
      [GeneseekDBofDBs_ProjectKB_01738]
      [GeneseekDBofDBs_ProjectKB_01739]
      [GeneseekDBofDBs_ProjectKB_01740]
      [GeneseekDBofDBs_ProjectKB_01741]
      [KolkerMediatedSchema_ProjectKB_Instance_223]
      [SNPTTest_ProjectKB_Instance_126]))

([GeneseekDBofDBs_ProjectKB_01736] of Boolean

    (boolean_value TRUE)
    (name "ButtonDisplayed-View Instance"))

([GeneseekDBofDBs_ProjectKB_01737] of Boolean

    (boolean_value FALSE)
    (name "ButtonDisplayed-References"))

([GeneseekDBofDBs_ProjectKB_01738] of Boolean

    (boolean_value TRUE)
    (name "ButtonDisplayed-Create Instance"))

([GeneseekDBofDBs_ProjectKB_01739] of Boolean

    (boolean_value TRUE)
    (name "ButtonDisplayed-Add"))

([GeneseekDBofDBs_ProjectKB_01740] of Boolean

    (boolean_value TRUE)
    (name "ButtonDisplayed-Remove"))

```

```

([GeneseekDBofDBs_ProjectKB_01741] of Boolean
  (boolean_value FALSE)
  (name "ButtonDisplayed-Delete Instance"))

([GeneseekDBofDBs_ProjectKB_01742] of Widget
  (height 60)
  (is_hidden FALSE)
  (name "__validation")
  (property_list [GeneseekDBofDBs_ProjectKB_01743])
  (widget_class_name
"edu.stanford.smi.protege.widget.ComboBoxWidget")
  (width 125)
  (x 0)
  (y 240))

([GeneseekDBofDBs_ProjectKB_01743] of Property_List
)

([GeneseekDBofDBs_ProjectKB_01744] of Widget
  (height 120)
  (is_hidden FALSE)
  (name "__maintenance")
  (property_list [GeneseekDBofDBs_ProjectKB_01745])
  (widget_class_name
"edu.stanford.smi.protege.widget.InstanceListWidget")
  (width 250)
  (x 250)
  (y 300))

([GeneseekDBofDBs_ProjectKB_01745] of Property_List
  (properties
    [GeneseekDBofDBs_ProjectKB_01746]
    [GeneseekDBofDBs_ProjectKB_01747]
    [GeneseekDBofDBs_ProjectKB_01748]
    [GeneseekDBofDBs_ProjectKB_01749]
    [GeneseekDBofDBs_ProjectKB_01750]
    [GeneseekDBofDBs_ProjectKB_01751]
    [KolkerMediatedSchema_ProjectKB_Instance_224]
    [KolkerMediatedSchema_ProjectKB_Instance_225]
    [SNPTest_ProjectKB_Instance_127]))

([GeneseekDBofDBs_ProjectKB_01746] of Boolean
  (boolean_value TRUE)
  (name "ButtonDisplayed-View Selected Instances"))

```

```

([GeneseekDBofDBs_ProjectKB_01747] of Boolean
  (boolean_value TRUE)
  (name "ButtonDisplayed-Create Instance"))

([GeneseekDBofDBs_ProjectKB_01748] of Boolean
  (boolean_value FALSE)
  (name "ButtonDisplayed-References"))

([GeneseekDBofDBs_ProjectKB_01749] of Boolean
  (boolean_value TRUE)
  (name "ButtonDisplayed-Add"))

([GeneseekDBofDBs_ProjectKB_01750] of Boolean
  (boolean_value TRUE)
  (name "ButtonDisplayed-Remove"))

([GeneseekDBofDBs_ProjectKB_01751] of Boolean
  (boolean_value FALSE)
  (name "ButtonDisplayed-Delete Selected Instances"))

([GeneseekDBofDBs_ProjectKB_01752] of Widget
  (height 60)
  (is_hidden FALSE)
  (name "__isCausal")
  (property_list [GeneseekDBofDBs_ProjectKB_01753])
  (widget_class_name
"edu.stanford.smi.protege.widget.CheckBoxWidget")
  (width 125)
  (x 0)
  (y 180))

([GeneseekDBofDBs_ProjectKB_01753] of Property_List
)

([GeneseekDBofDBs_ProjectKB_02121] of Widget
  (height 605)
  (is_hidden FALSE)
  (name "SourceIDPair")
  (property_list [GeneseekDBofDBs_ProjectKB_02122])
  (widget_class_name
"edu.stanford.smi.protege.widget.FormWidget")
  (width 746)
  (x 0)
  (y 0))

```

```

([GeneseekDBofDBs_ProjectKB_02122] of Property_List

  (properties
    [GeneseekDBofDBs_ProjectKB_02123]
    [GeneseekDBofDBs_ProjectKB_02131]))

([GeneseekDBofDBs_ProjectKB_02123] of Widget

  (height 60)
  (is_hidden FALSE)
  (name "DB")
  (property_list [GeneseekDBofDBs_ProjectKB_02124])
  (widget_class_name
"edu.stanford.smi.protege.widget.InstanceFieldWidget")
  (width 250)
  (x 0)
  (y 60))

([GeneseekDBofDBs_ProjectKB_02124] of Property_List

  (properties
    [GeneseekDBofDBs_ProjectKB_02125]
    [GeneseekDBofDBs_ProjectKB_02126]
    [GeneseekDBofDBs_ProjectKB_02127]
    [GeneseekDBofDBs_ProjectKB_02128]
    [GeneseekDBofDBs_ProjectKB_02129]
    [GeneseekDBofDBs_ProjectKB_02130]))

([GeneseekDBofDBs_ProjectKB_02125] of Boolean

  (boolean_value TRUE)
  (name "ButtonDisplayed-View Instance"))

([GeneseekDBofDBs_ProjectKB_02126] of Boolean

  (boolean_value FALSE)
  (name "ButtonDisplayed-References"))

([GeneseekDBofDBs_ProjectKB_02127] of Boolean

  (boolean_value TRUE)
  (name "ButtonDisplayed-Create Instance"))

([GeneseekDBofDBs_ProjectKB_02128] of Boolean

  (boolean_value TRUE)
  (name "ButtonDisplayed-Add"))

([GeneseekDBofDBs_ProjectKB_02129] of Boolean

```

```

        (boolean_value TRUE)
        (name "ButtonDisplayed-Remove"))

    ([GeneseekDBofDBs_ProjectKB_02130] of Boolean

        (boolean_value FALSE)
        (name "ButtonDisplayed-Delete Instance"))

    ([GeneseekDBofDBs_ProjectKB_02131] of Widget

        (height 60)
        (is_hidden FALSE)
        (name "ID")
        (property_list [GeneseekDBofDBs_ProjectKB_02132])
        (widget_class_name
"edu.stanford.smi.protege.widget.TextFieldWidget")
        (width 250)
        (x 0)
        (y 0))

    ([GeneseekDBofDBs_ProjectKB_02132] of Property_List
)

    ([GeneseekDBofDBs_ProjectKB_02157] of Widget

        (height 1010)
        (is_hidden FALSE)
        (name "RelationshipMapping")
        (property_list [GeneseekDBofDBs_ProjectKB_01500])
        (widget_class_name
"edu.stanford.smi.protege.widget.FormWidget")
        (width 1320)
        (x 0)
        (y 0))

    ([GeneseekDBofDBs_ProjectKB_02265] of Widget

        (height 605)
        (is_hidden FALSE)
        (name "NucleotideSequence")
        (property_list [GeneseekDBofDBs_ProjectKB_02280])
        (widget_class_name
"edu.stanford.smi.protege.widget.FormWidget")
        (width 746)
        (x 0)
        (y 0))

    ([GeneseekDBofDBs_ProjectKB_02280] of Property_List

        (name "class widget properties")
        (properties

```

```

[GeneseekDBofDBs_ProjectKB_02281]
[GeneseekDBofDBs_ProjectKB_02284]
[GeneseekDBofDBs_ProjectKB_02289]))

([GeneseekDBofDBs_ProjectKB_02281] of Property_List
  (name "layout properties")
  (properties
    [GeneseekDBofDBs_ProjectKB_02282]
    [GeneseekDBofDBs_ProjectKB_02283]))

([GeneseekDBofDBs_ProjectKB_02282] of String
  (name "horizontal_stretcher"))

([GeneseekDBofDBs_ProjectKB_02283] of String
  (name "vertical_stretcher"))

([GeneseekDBofDBs_ProjectKB_02284] of Widget
  (height 120)
  (is_hidden FALSE)
  (name "Name")
  (property_list [GeneseekDBofDBs_ProjectKB_02285])
  (widget_class_name
"edu.stanford.smi.protege.widget.StringListWidget")
  (width 250)
  (x 0)
  (y 0))

([GeneseekDBofDBs_ProjectKB_02285] of Property_List
  (properties
    [GeneseekDBofDBs_ProjectKB_02286]
    [GeneseekDBofDBs_ProjectKB_02287]
    [GeneseekDBofDBs_ProjectKB_02288]))

([GeneseekDBofDBs_ProjectKB_02286] of Boolean
  (boolean_value TRUE)
  (name "ButtonDisplayed-Edit the selected strings"))

([GeneseekDBofDBs_ProjectKB_02287] of Boolean
  (boolean_value TRUE)
  (name "ButtonDisplayed-Create a new string"))

([GeneseekDBofDBs_ProjectKB_02288] of Boolean
  (boolean_value TRUE)

```

```

      (name "ButtonDisplayed-Remove"))

  ([GeneseekDBofDBs_ProjectKB_02289] of Widget

    (height 60)
    (is_hidden FALSE)
    (name "Source")
    (property_list [GeneseekDBofDBs_ProjectKB_02290])
    (widget_class_name
"edu.stanford.smi.protege.widget.InstanceFieldWidget")
    (width 250)
    (x 0)
    (y 120))

  ([GeneseekDBofDBs_ProjectKB_02290] of Property_List

    (properties
      [GeneseekDBofDBs_ProjectKB_02291]
      [GeneseekDBofDBs_ProjectKB_02292]
      [GeneseekDBofDBs_ProjectKB_02293]
      [GeneseekDBofDBs_ProjectKB_02294]
      [GeneseekDBofDBs_ProjectKB_02295]
      [GeneseekDBofDBs_ProjectKB_02296]))

  ([GeneseekDBofDBs_ProjectKB_02291] of Boolean

    (boolean_value TRUE)
    (name "ButtonDisplayed-View Instance"))

  ([GeneseekDBofDBs_ProjectKB_02292] of Boolean

    (boolean_value FALSE)
    (name "ButtonDisplayed-References"))

  ([GeneseekDBofDBs_ProjectKB_02293] of Boolean

    (boolean_value TRUE)
    (name "ButtonDisplayed-Create Instance"))

  ([GeneseekDBofDBs_ProjectKB_02294] of Boolean

    (boolean_value TRUE)
    (name "ButtonDisplayed-Add"))

  ([GeneseekDBofDBs_ProjectKB_02295] of Boolean

    (boolean_value TRUE)
    (name "ButtonDisplayed-Remove"))

  ([GeneseekDBofDBs_ProjectKB_02296] of Boolean

```

```

        (boolean_value FALSE)
        (name "ButtonDisplayed-Delete Instance"))

    ([GeneseekDBofDBs_ProjectKB_02297] of Widget

        (height 605)
        (is_hidden FALSE)
        (name "Gene")
        (property_list [GeneseekDBofDBs_ProjectKB_02314])
        (widget_class_name
"edu.stanford.smi.protege.widget.FormWidget")
        (width 746)
        (x 0)
        (y 0))

    ([GeneseekDBofDBs_ProjectKB_02314] of Property_List

        (name "class widget properties")
        (properties
            [GeneseekDBofDBs_ProjectKB_02315]
            [GeneseekDBofDBs_ProjectKB_02318]
            [GeneseekDBofDBs_ProjectKB_02323]
            [GeneseekDBofDBs_ProjectKB_02331]))

    ([GeneseekDBofDBs_ProjectKB_02315] of Property_List

        (name "layout properties")
        (properties
            [GeneseekDBofDBs_ProjectKB_02316]
            [GeneseekDBofDBs_ProjectKB_02317]))

    ([GeneseekDBofDBs_ProjectKB_02316] of String

        (name "horizontal_stretcher"))

    ([GeneseekDBofDBs_ProjectKB_02317] of String

        (name "vertical_stretcher"))

    ([GeneseekDBofDBs_ProjectKB_02318] of Widget

        (height 120)
        (is_hidden FALSE)
        (name "Name")
        (property_list [GeneseekDBofDBs_ProjectKB_02319])
        (widget_class_name
"edu.stanford.smi.protege.widget.StringListWidget")
        (width 250)
        (x 0)
        (y 0))

```

```

([GeneseekDBofDBs_ProjectKB_02319] of Property_List

  (properties
    [GeneseekDBofDBs_ProjectKB_02320]
    [GeneseekDBofDBs_ProjectKB_02321]
    [GeneseekDBofDBs_ProjectKB_02322]))

([GeneseekDBofDBs_ProjectKB_02320] of Boolean

  (boolean_value TRUE)
  (name "ButtonDisplayed-Edit the selected strings"))

([GeneseekDBofDBs_ProjectKB_02321] of Boolean

  (boolean_value TRUE)
  (name "ButtonDisplayed-Create a new string"))

([GeneseekDBofDBs_ProjectKB_02322] of Boolean

  (boolean_value TRUE)
  (name "ButtonDisplayed-Remove"))

([GeneseekDBofDBs_ProjectKB_02323] of Widget

  (height 60)
  (is_hidden FALSE)
  (name "Source")
  (property_list [GeneseekDBofDBs_ProjectKB_02324])
  (widget_class_name
"edu.stanford.smi.protege.widget.InstanceFieldWidget")
  (width 250)
  (x 0)
  (y 120))

([GeneseekDBofDBs_ProjectKB_02324] of Property_List

  (properties
    [GeneseekDBofDBs_ProjectKB_02325]
    [GeneseekDBofDBs_ProjectKB_02326]
    [GeneseekDBofDBs_ProjectKB_02327]
    [GeneseekDBofDBs_ProjectKB_02328]
    [GeneseekDBofDBs_ProjectKB_02329]
    [GeneseekDBofDBs_ProjectKB_02330]))

([GeneseekDBofDBs_ProjectKB_02325] of Boolean

  (boolean_value TRUE)
  (name "ButtonDisplayed-View Instance"))

([GeneseekDBofDBs_ProjectKB_02326] of Boolean

```

```

      (boolean_value FALSE)
      (name "ButtonDisplayed-References"))

  ([GeneseekDBofDBs_ProjectKB_02327] of Boolean
    (boolean_value TRUE)
    (name "ButtonDisplayed-Create Instance"))

  ([GeneseekDBofDBs_ProjectKB_02328] of Boolean
    (boolean_value TRUE)
    (name "ButtonDisplayed-Add"))

  ([GeneseekDBofDBs_ProjectKB_02329] of Boolean
    (boolean_value TRUE)
    (name "ButtonDisplayed-Remove"))

  ([GeneseekDBofDBs_ProjectKB_02330] of Boolean
    (boolean_value FALSE)
    (name "ButtonDisplayed-Delete Instance"))

  ([GeneseekDBofDBs_ProjectKB_02331] of Widget
    (height 60)
    (is_hidden FALSE)
    (name "Locus")
    (property_list [GeneseekDBofDBs_ProjectKB_02332])
    (widget_class_name
"edu.stanford.smi.protege.widget.TextFieldWidget")
    (width 250)
    (x 0)
    (y 180))

  ([GeneseekDBofDBs_ProjectKB_02332] of Property_List
)

  ([GeneseekDBofDBs_ProjectKB_02365] of Widget
    (height 605)
    (is_hidden FALSE)
    (name "Protein")
    (property_list [GeneseekDBofDBs_ProjectKB_02380])
    (widget_class_name
"edu.stanford.smi.protege.widget.FormWidget")
    (width 746)
    (x 0)
    (y 0))

  ([GeneseekDBofDBs_ProjectKB_02380] of Property_List

```

```

(name "class widget properties")
(properties
  [GeneseekDBofDBs_ProjectKB_02381]
  [GeneseekDBofDBs_ProjectKB_02384]
  [GeneseekDBofDBs_ProjectKB_02389]))

([GeneseekDBofDBs_ProjectKB_02381] of Property_List

  (name "layout properties")
  (properties
    [GeneseekDBofDBs_ProjectKB_02382]
    [GeneseekDBofDBs_ProjectKB_02383]))

([GeneseekDBofDBs_ProjectKB_02382] of String

  (name "horizontal_stretcher"))

([GeneseekDBofDBs_ProjectKB_02383] of String

  (name "vertical_stretcher"))

([GeneseekDBofDBs_ProjectKB_02384] of Widget

  (height 120)
  (is_hidden FALSE)
  (name "Name")
  (property_list [GeneseekDBofDBs_ProjectKB_02385])
  (widget_class_name
"edu.stanford.smi.protege.widget.StringListWidget")
  (width 250)
  (x 0)
  (y 0))

([GeneseekDBofDBs_ProjectKB_02385] of Property_List

  (properties
    [GeneseekDBofDBs_ProjectKB_02386]
    [GeneseekDBofDBs_ProjectKB_02387]
    [GeneseekDBofDBs_ProjectKB_02388]))

([GeneseekDBofDBs_ProjectKB_02386] of Boolean

  (boolean_value TRUE)
  (name "ButtonDisplayed-Edit the selected strings"))

([GeneseekDBofDBs_ProjectKB_02387] of Boolean

  (boolean_value TRUE)
  (name "ButtonDisplayed-Create a new string"))

```

```

([GeneseekDBofDBs_ProjectKB_02388] of Boolean
  (boolean_value TRUE)
  (name "ButtonDisplayed-Remove"))

([GeneseekDBofDBs_ProjectKB_02389] of Widget
  (height 60)
  (is_hidden FALSE)
  (name "Source")
  (property_list [GeneseekDBofDBs_ProjectKB_02390])
  (widget_class_name
"edu.stanford.smi.protege.widget.InstanceFieldWidget")
  (width 250)
  (x 0)
  (y 120))

([GeneseekDBofDBs_ProjectKB_02390] of Property_List
  (properties
    [GeneseekDBofDBs_ProjectKB_02391]
    [GeneseekDBofDBs_ProjectKB_02392]
    [GeneseekDBofDBs_ProjectKB_02393]
    [GeneseekDBofDBs_ProjectKB_02394]
    [GeneseekDBofDBs_ProjectKB_02395]
    [GeneseekDBofDBs_ProjectKB_02396]))

([GeneseekDBofDBs_ProjectKB_02391] of Boolean
  (boolean_value TRUE)
  (name "ButtonDisplayed-View Instance"))

([GeneseekDBofDBs_ProjectKB_02392] of Boolean
  (boolean_value FALSE)
  (name "ButtonDisplayed-References"))

([GeneseekDBofDBs_ProjectKB_02393] of Boolean
  (boolean_value TRUE)
  (name "ButtonDisplayed-Create Instance"))

([GeneseekDBofDBs_ProjectKB_02394] of Boolean
  (boolean_value TRUE)
  (name "ButtonDisplayed-Add"))

([GeneseekDBofDBs_ProjectKB_02395] of Boolean
  (boolean_value TRUE)
  (name "ButtonDisplayed-Remove"))

```

```

([GeneseekDBofDBs_ProjectKB_02396] of Boolean
  (boolean_value FALSE)
  (name "ButtonDisplayed-Delete Instance"))

([GeneseekDBofDBs_ProjectKB_Instance_130] of Widget
  (is_hidden FALSE)
  (name "Supported")
  (property_list [GeneseekDBofDBs_ProjectKB_Instance_133])
  (widget_class_name
"edu.stanford.smi.protege.widget.FormWidget"))

([GeneseekDBofDBs_ProjectKB_Instance_133] of Property_List
  (properties
    [GeneseekDBofDBs_ProjectKB_Instance_134]
    [GeneseekDBofDBs_ProjectKB_Instance_136]
    [GeneseekDBofDBs_ProjectKB_Instance_137]
    [GeneseekDBofDBs_ProjectKB_Instance_145]
    [GeneseekDBofDBs_ProjectKB_Instance_150]
    [GeneseekDBofDBs_ProjectKB_Instance_155]))

([GeneseekDBofDBs_ProjectKB_Instance_134] of Widget
  (height 60)
  (is_hidden FALSE)
  (name "DBName")
  (property_list [GeneseekDBofDBs_ProjectKB_Instance_135])
  (widget_class_name
"edu.stanford.smi.protege.widget.TextFieldWidget")
  (width 250)
  (x 0)
  (y 0))

([GeneseekDBofDBs_ProjectKB_Instance_135] of Property_List
)

([GeneseekDBofDBs_ProjectKB_Instance_136] of Property_List
  (name "layout properties"))

([GeneseekDBofDBs_ProjectKB_Instance_137] of Widget
  (height 180)
  (is_hidden FALSE)
  (name "DBContains")
  (property_list [GeneseekDBofDBs_ProjectKB_Instance_138])
  (widget_class_name
"edu.stanford.smi.protege.widget.InstanceListWidget")

```

```

(width 250)
(x 260)
(y 0))

([GeneseekDBofDBs_ProjectKB_Instance_138] of Property_List

(properties
  [GeneseekDBofDBs_ProjectKB_Instance_139]
  [GeneseekDBofDBs_ProjectKB_Instance_140]
  [GeneseekDBofDBs_ProjectKB_Instance_141]
  [GeneseekDBofDBs_ProjectKB_Instance_142]
  [GeneseekDBofDBs_ProjectKB_Instance_143]
  [GeneseekDBofDBs_ProjectKB_Instance_144]
  [KolkerMediatedSchema_ProjectKB_Instance_105]
  [KolkerMediatedSchema_ProjectKB_Instance_106]
  [SNPTTest_ProjectKB_Instance_141]))

([GeneseekDBofDBs_ProjectKB_Instance_139] of Boolean

(boolean_value TRUE)
(name "ButtonDisplayed-View Selected Instances"))

([GeneseekDBofDBs_ProjectKB_Instance_140] of Boolean

(boolean_value TRUE)
(name "ButtonDisplayed-Create Instance"))

([GeneseekDBofDBs_ProjectKB_Instance_141] of Boolean

(boolean_value FALSE)
(name "ButtonDisplayed-References"))

([GeneseekDBofDBs_ProjectKB_Instance_142] of Boolean

(boolean_value TRUE)
(name "ButtonDisplayed-Add"))

([GeneseekDBofDBs_ProjectKB_Instance_143] of Boolean

(boolean_value TRUE)
(name "ButtonDisplayed-Remove"))

([GeneseekDBofDBs_ProjectKB_Instance_144] of Boolean

(boolean_value FALSE)
(name "ButtonDisplayed-Delete Selected Instances"))

([GeneseekDBofDBs_ProjectKB_Instance_145] of Widget

(height 120)
(is_hidden FALSE)

```

```

        (name "DBPrimaryEntity")
        (property_list [GeneseekDBofDBs_ProjectKB_Instance_146])
        (widget_class_name
"edu.stanford.smi.protege.widget.ClsListWidget")
        (width 250)
        (x 0)
        (y 60))

([GeneseekDBofDBs_ProjectKB_Instance_146] of Property_List

  (properties
    [GeneseekDBofDBs_ProjectKB_Instance_147]
    [GeneseekDBofDBs_ProjectKB_Instance_148]
    [GeneseekDBofDBs_ProjectKB_Instance_149]))

([GeneseekDBofDBs_ProjectKB_Instance_147] of Boolean

  (boolean_value TRUE)
  (name "ButtonDisplayed-View Selected Classes"))

([GeneseekDBofDBs_ProjectKB_Instance_148] of Boolean

  (boolean_value TRUE)
  (name "ButtonDisplayed-Add"))

([GeneseekDBofDBs_ProjectKB_Instance_149] of Boolean

  (boolean_value TRUE)
  (name "ButtonDisplayed-Remove"))

([GeneseekDBofDBs_ProjectKB_Instance_150] of Widget

  (height 120)
  (is_hidden FALSE)
  (name "DBMapping")
  (property_list [GeneseekDBofDBs_ProjectKB_Instance_151])
  (widget_class_name
"edu.stanford.smi.protege.widget.StringListWidget")
  (width 250)
  (x 0)
  (y 180))

([GeneseekDBofDBs_ProjectKB_Instance_151] of Property_List

  (properties
    [GeneseekDBofDBs_ProjectKB_Instance_152]
    [GeneseekDBofDBs_ProjectKB_Instance_153]
    [GeneseekDBofDBs_ProjectKB_Instance_154]))

([GeneseekDBofDBs_ProjectKB_Instance_152] of Boolean

```

```

    (boolean_value TRUE)
    (name "ButtonDisplayed-Edit the selected strings"))

  ([[GeneseekDBofDBs_ProjectKB_Instance_153] of Boolean

    (boolean_value TRUE)
    (name "ButtonDisplayed-Create a new string"))

  ([[GeneseekDBofDBs_ProjectKB_Instance_154] of Boolean

    (boolean_value TRUE)
    (name "ButtonDisplayed-Remove"))

  ([[GeneseekDBofDBs_ProjectKB_Instance_155] of Widget

    (height 230)
    (is_hidden FALSE)
    (name "Notes")
    (property_list [GeneseekDBofDBs_ProjectKB_Instance_156])
    (widget_class_name
"edu.stanford.smi.protege.widget.TextAreaWidget")
    (width 510)
    (x 0)
    (y 330))

  ([[GeneseekDBofDBs_ProjectKB_Instance_156] of Property_List
)

  ([[INSTANCES_TAB] of Widget

    (is_hidden FALSE)
    (label "Instances")
    (property_list [ProjectKB_0212_00037])
    (widget_class_name
"edu.stanford.smi.protege.widget.InstancesTab"))

  ([[KB_862621_Instance_0] of Property_List

    (name "class widget properties")
    (properties
      [KB_862621_Instance_1]
      [KB_862621_Instance_2]
      [KB_862621_Instance_3]
      [KB_862621_Instance_4]
      [KB_862621_Instance_5]
      [KB_862621_Instance_6]
      [KB_862621_Instance_7]
      [KB_862621_Instance_8]
      [KB_862621_Instance_9]
      [KB_862621_Instance_10]))

```

```

([KB_862621_Instance_1] of Widget

  (height 120)
  (label "Constraints")
  (name ":SLOT-CONSTRAINTS")
  (property_list [SNPTest_ProjectKB_Instance_10109])
  (widget_class_name
"edu.stanford.smi.protege.widget.ConstraintsWidget")
  (width 200)
  (x 400)
  (y 0))

([KB_862621_Instance_10] of Property_List

  (name "layout properties")
  (properties [KB_862621_Instance_11]))

([KB_862621_Instance_11] of String

  (name "vertical_stretcher")
  (string_value ":DIRECT-TEMPLATE-SLOTS"))

([KB_862621_Instance_12] of Property_List

  (name "slot widget properties")
  (properties
    [KB_862621_Instance_13]
    [KB_862621_Instance_14]
    [KB_862621_Instance_15]
    [KB_862621_Instance_16]
    [KB_862621_Instance_17]
    [KB_862621_Instance_18]
    [KB_862621_Instance_19]
    [KB_862621_Instance_20]
    [KB_862621_Instance_21]
    [KB_862621_Instance_22]
    [KB_862621_Instance_23]
    [KB_862621_Instance_24]
    [KB_862621_Instance_25]
    [KB_862621_Instance_26]
    [KB_862621_Instance_27]
    [KB_862621_Instance_28]
    [SNPTest_ProjectKB_Instance_10118]))

([KB_862621_Instance_13] of Widget

  (height 60)
  (label "Cardinality")
  (name ":SLOT-MINIMUM-CARDINALITY")
  (property_list [SNPTest_ProjectKB_Instance_10126])

```

```

        (widget_class_name
"edu.stanford.smi.protege.widget.MinimumCardinalityWidget")
        (width 200)
        (x 200)
        (y 120))

([KB_862621_Instance_14] of Widget

        (height 35)
        (name ":SLOT-MAXIMUM-CARDINALITY")
        (property_list [SNPTest_ProjectKB_Instance_10127])
        (widget_class_name
"edu.stanford.smi.protege.widget.MaximumCardinalityWidget")
        (width 200)
        (x 200)
        (y 180))

([KB_862621_Instance_15] of Widget

        (name ":SLOT-CONSTRAINTS"))

([KB_862621_Instance_16] of Widget

        (name ":DIRECT-TYPE"))

([KB_862621_Instance_17] of Widget

        (height 95)
        (label "Domain")
        (name ":DIRECT-DOMAIN")
        (property_list [SNPTest_ProjectKB_Instance_10119])
        (widget_class_name
"edu.stanford.smi.protege.widget.DirectDomainWidget")
        (width 200)
        (x 400)
        (y 180))

([KB_862621_Instance_18] of Widget

        (height 90)
        (label "Template Values")
        (name ":SLOT-VALUES")
        (property_list [SNPTest_ProjectKB_Instance_10120])
        (widget_class_name
"edu.stanford.smi.protege.widget.SlotValuesWidget")
        (width 200)
        (x 400)
        (y 0))

([KB_862621_Instance_19] of Widget

```

```

        (name ":DIRECT-SUPERSLOTS"))
      ([KB_862621_Instance_2] of Widget
        (name ":DIRECT-INSTANCES"))
      ([KB_862621_Instance_20] of Widget
        (name ":DIRECT-SUBSLOTS"))
      ([KB_862621_Instance_21] of Widget
        (height 90)
        (label "Default")
        (name ":SLOT-DEFAULTS")
        (property_list [SNPTest_ProjectKB_Instance_10121])
        (widget_class_name
"edu.stanford.smi.protege.widget.DefaultValuesWidget")
        (width 200)
        (x 400)
        (y 90))
      ([KB_862621_Instance_22] of Widget
        (height 120)
        (label "Documentation")
        (name ":DOCUMENTATION")
        (property_list [SNPTest_ProjectKB_Instance_10122])
        (widget_class_name
"edu.stanford.smi.protege.widget.DocumentationWidget")
        (width 200)
        (x 200)
        (y 0))
      ([KB_862621_Instance_23] of Widget
        (height 60)
        (label "Maximum")
        (name ":SLOT-NUMERIC-MAXIMUM")
        (property_list [SNPTest_ProjectKB_Instance_10123])
        (widget_class_name
"edu.stanford.smi.protege.widget.NumericMaximumWidget")
        (width 100)
        (x 100)
        (y 215))
      ([KB_862621_Instance_24] of Widget
        (height 60)
        (label "Minimum")
        (name ":SLOT-NUMERIC-MINIMUM"))

```

```

        (property_list [SNPTest_ProjectKB_Instance_10124])
        (widget_class_name
"edu.stanford.smi.protege.widget.NumericMinimumWidget")
        (width 100)
        (x 0)
        (y 215))

([KB_862621_Instance_25] of Widget

        (name ":ASSOCIATED-FACET"))

([KB_862621_Instance_26] of Widget

        (height 60)
        (label "Name")
        (name ":NAME")
        (property_list [SNPTest_ProjectKB_Instance_10125])
        (widget_class_name
"edu.stanford.smi.protege.widget.InstanceNameWidget")
        (width 200)
        (x 0)
        (y 0))

([KB_862621_Instance_27] of Widget

        (height 60)
        (label "Inverse Slot")
        (name ":SLOT-INVERSE")
        (property_list [SNPTest_ProjectKB_Instance_10128])
        (widget_class_name
"edu.stanford.smi.protege.widget.InverseSlotWidget")
        (width 200)
        (x 200)
        (y 215))

([KB_862621_Instance_28] of Widget

        (height 155)
        (label "Value Type")
        (name ":SLOT-VALUE-TYPE")
        (property_list [SNPTest_ProjectKB_Instance_10129])
        (widget_class_name
"edu.stanford.smi.protege.widget.ValueTypeWidget")
        (width 200)
        (x 0)
        (y 60))

([KB_862621_Instance_29] of Property_List

        (name "facet widget properties")
        (properties

```

```

[KB_862621_Instance_30]
[KB_862621_Instance_31]
[KB_862621_Instance_32]))

([KB_862621_Instance_3] of Widget
  (name ":DIRECT-SUBCLASSES"))

([KB_862621_Instance_30] of Widget
  (height 60)
  (label "Name")
  (name ":NAME")
  (widget_class_name
"edu.stanford.smi.protege.widget.InstanceNameWidget")
  (width 200)
  (x 0)
  (y 0))

([KB_862621_Instance_31] of Widget
  (height 120)
  (label "Documentation")
  (name ":DOCUMENTATION")
  (widget_class_name
"edu.stanford.smi.protege.widget.DocumentationWidget")
  (width 200)
  (x 200)
  (y 0))

([KB_862621_Instance_32] of Widget
  (height 60)
  (label "Associated Slot")
  (name ":ASSOCIATED-SLOT")
  (widget_class_name
"edu.stanford.smi.protege.widget.InstanceFieldWidget")
  (width 200)
  (x 0)
  (y 60))

([KB_862621_Instance_33] of Property_List
  (properties
    [KB_862621_Instance_34]
    [KB_862621_Instance_35]
    [KB_862621_Instance_36]
    [KB_862621_Instance_37]))

([KB_862621_Instance_34] of Widget

```

```

      (name ":ANNOTATED-INSTANCE"))
    ([KB_862621_Instance_35] of Widget
      (name ":CREATOR"))
    ([KB_862621_Instance_36] of Widget
      (name ":CREATION-TIMESTAMP"))
    ([KB_862621_Instance_37] of Widget
      (height 100)
      (is_hidden FALSE)
      (name ":ANNOTATION-TEXT")
      (widget_class_name
"edu.stanford.smi.protege.widget.YellowStickyWidget")
      (width 200)
      (x 0)
      (y 0))
    ([KB_862621_Instance_38] of Property_List
      (properties
        [KB_862621_Instance_39]
        [KB_862621_Instance_40]
        [KB_862621_Instance_41]
        [KB_862621_Instance_42]))
    ([KB_862621_Instance_39] of Widget
      (height 60)
      (is_hidden FALSE)
      (label "Name")
      (name ":PAL-NAME")
      (widget_class_name
"edu.stanford.smi.protege.widget.TextFieldWidget")
      (width 275)
      (x 0)
      (y 0))
    ([KB_862621_Instance_4] of Widget
      (name ":DIRECT-SUPERCLASSES"))
    ([KB_862621_Instance_40] of Widget
      (height 180)
      (is_hidden FALSE)
      (label "Range")
      (name ":PAL-RANGE"))

```

```

        (widget_class_name
"edu.stanford.smi.protege.widget.pal.constraint.PalRangeWidget")
        (width 250)
        (x 275)
        (y 180))

([KB_862621_Instance_41] of Widget

        (height 180)
        (is_hidden FALSE)
        (label "Description")
        (name ":PAL-DESCRIPTION")
        (widget_class_name
"edu.stanford.smi.protege.widget.TextAreaWidget")
        (width 250)
        (x 275)
        (y 0))

([KB_862621_Instance_42] of Widget

        (height 300)
        (is_hidden FALSE)
        (label "Statement")
        (name ":PAL-STATEMENT")
        (widget_class_name
"edu.stanford.smi.protege.widget.pal.constraint.PalConstraintWidget
")
        (width 275)
        (x 0)
        (y 60))

([KB_862621_Instance_5] of Widget

        (height 120)
        (label "Documentation")
        (name ":DOCUMENTATION")
        (property_list [SNPTest_ProjectKB_Instance_10112])
        (widget_class_name
"edu.stanford.smi.protege.widget.DocumentationWidget")
        (width 200)
        (x 200)
        (y 0))

([KB_862621_Instance_6] of Widget

        (height 60)
        (label "Name")
        (name ":NAME")
        (property_list [SNPTest_ProjectKB_Instance_10113])
        (widget_class_name
"edu.stanford.smi.protege.widget.InstanceNameWidget")

```

```

        (width 200)
        (x 0)
        (y 0))

([KB_862621_Instance_7] of Widget

    (height 60)
    (label "Role")
    (name ":ROLE")
    (property_list [SNPTest_ProjectKB_Instance_10117])
    (widget_class_name
"edu.stanford.smi.protege.widget.RoleWidget")
    (width 200)
    (x 0)
    (y 60))

([KB_862621_Instance_8] of Widget

    (name ":DIRECT-TYPE"))

([KB_862621_Instance_9] of Widget

    (height 150)
    (label "Template Slots")
    (name ":DIRECT-TEMPLATE-SLOTS")
    (property_list [SNPTest_ProjectKB_Instance_10114])
    (widget_class_name
"edu.stanford.smi.protege.widget.TemplateSlotsWidget")
    (width 600)
    (x 0)
    (y 120))

([KolkerMediatedSchema_ProjectKB_Instance_105] of Boolean

    (boolean_value FALSE)
    (name "ButtonDisplayed-View References to Value "))

([KolkerMediatedSchema_ProjectKB_Instance_106] of Boolean

    (boolean_value FALSE)
    (name "ButtonDisplayed-Delete Instance"))

([KolkerMediatedSchema_ProjectKB_Instance_107] of Boolean

    (boolean_value FALSE)
    (name "ButtonDisplayed-View References to Value "))

([KolkerMediatedSchema_ProjectKB_Instance_108] of Boolean

    (boolean_value FALSE)
    (name "ButtonDisplayed-Delete Instance"))

```

```

([KolkerMediatedSchema_ProjectKB_Instance_221] of Boolean
  (boolean_value FALSE)
  (name "ButtonDisplayed-View References to Value "))

([KolkerMediatedSchema_ProjectKB_Instance_222] of Boolean
  (boolean_value FALSE)
  (name "ButtonDisplayed-View References to Value "))

([KolkerMediatedSchema_ProjectKB_Instance_223] of Boolean
  (boolean_value FALSE)
  (name "ButtonDisplayed-View References to Value "))

([KolkerMediatedSchema_ProjectKB_Instance_224] of Boolean
  (boolean_value FALSE)
  (name "ButtonDisplayed-View References to Value "))

([KolkerMediatedSchema_ProjectKB_Instance_225] of Boolean
  (boolean_value FALSE)
  (name "ButtonDisplayed-Delete Instance"))

([KolkerMediatedSchema_ProjectKB_Instance_80] of Boolean
  (boolean_value FALSE)
  (name "ButtonDisplayed-View References to Value "))

([KolkerMediatedSchema_ProjectKB_Instance_81] of Boolean
  (boolean_value FALSE)
  (name "ButtonDisplayed-Delete Instance"))

([KolkerMediatedSchema_ProjectKB_Instance_82] of Boolean
  (boolean_value FALSE)
  (name "ButtonDisplayed-View References to Value "))

([KolkerTest_ProjectKB_Instance_10012] of Widget
  (height 60)
  (is_hidden FALSE)
  (name "Qs")
  (property_list [KolkerTest_ProjectKB_Instance_10013])
  (widget_class_name
"edu.stanford.smi.protege.widget.FloatFieldWidget")
  (width 100)
  (x 0)

```

```

(y 420))

([KolkerTest_ProjectKB_Instance_10013] of Property_List
)

([KolkerTest_ProjectKB_Instance_20039] of Widget

  (height 60)
  (is_hidden FALSE)
  (name "Ps")
  (property_list [KolkerTest_ProjectKB_Instance_20040])
  (widget_class_name
"edu.stanford.smi.protege.widget.FloatFieldWidget")
  (width 100)
  (x 0)
  (y 180))

([KolkerTest_ProjectKB_Instance_20040] of Property_List
)

([PAL_FORM_WIDGET] of Widget

  (height 597)
  (is_hidden FALSE)
  (name ":PAL-CONSTRAINT")
  (property_list [KB_862621_Instance_38])
  (widget_class_name
"edu.stanford.smi.protege.widget.FormWidget")
  (width 738)
  (x 0)
  (y 0))

([PROJECT] of Project

  (browser_slot_names [BROWSER_SLOT_NAMES])
  (customized_instance_widgets
    [STANDARD_SLOT_FORM_WIDGET]
    [GeneseekDBofDBs_ProjectKB_02265]
    [GeneseekDBofDBs_ProjectKB_00361]
    [GeneseekDBofDBs_ProjectKB_00302]
    [GeneseekDBofDBs_ProjectKB_00256]
    [GeneseekDBofDBs_ProjectKB_02297]
    [PAL_FORM_WIDGET]
    [GeneseekDBofDBs_ProjectKB_02365]
    [GeneseekDBofDBs_ProjectKB_Instance_130]
    [GeneseekDBofDBs_ProjectKB_00679]
    [STANDARD_CLASS_FORM_WIDGET]
    [GeneseekDBofDBs_ProjectKB_00204]
    [GeneseekDBofDBs_ProjectKB_01185]
    [GeneseekDBofDBs_ProjectKB_02121]
    [GeneseekDBofDBs_ProjectKB_00596]
  )
)

```

```

        [GeneseekDBofDBs_ProjectKB_00209]
        [GeneseekDBofDBs_ProjectKB_02157])
    (default_cls_metaclass ":STANDARD-CLASS")
    (default_facet_metaclass ":STANDARD-FACET")
    (default_instance_widget_class_name
"edu.stanford.smi.protege.widget.FormWidget")
    (default_slot_metaclass ":STANDARD-SLOT")
    (journaling_enabled FALSE)
    (next_frame_number 0)
    (options [ProjectKB_0212_00006])
    (property_map [GeneseekDBofDBs_ProjectKB_00640])
    (sources [SOURCES])
    (tabs
        [CLSES_TAB]
        [SLOTS_TAB]
        [FORMS_TAB]
        [INSTANCES_TAB]
        [QUERIES_TAB]
        [ProjectKB_0212_00003]
        [ProjectKB_0212_00001]))

    ([ProjectKB_0212_00000] of String
        (name "factory_class_name")
        (string_value
"edu.stanford.smi.protege.storage.clips.ClipsKnowledgeBaseFactory"))

    ([ProjectKB_0212_00001] of Widget
        (is_hidden TRUE)
        (property_list [ProjectKB_0212_00002])
        (widget_class_name
"edu.stanford.smi.protege.widget.KAToolTab"))

    ([ProjectKB_0212_00002] of Property_List
    )

    ([ProjectKB_0212_00003] of Widget
        (is_hidden FALSE)
        (property_list [ProjectKB_0212_00004])
        (widget_class_name
"edu.stanford.smi.protege.widget.ClsesAndInstancesTab"))

    ([ProjectKB_0212_00004] of Property_List
    )

    ([ProjectKB_0212_00005] of Property_List
        (properties
            [GeneseekDBofDBs_ProjectKB_00041])

```

```

        [GeneseekDBofDBs_ProjectKB_00042]))

([ProjectKB_0212_00006] of Options

    (confirm_on_remove FALSE)
    (display_abstract_class_icon TRUE).
    (display_hidden_classes TRUE)
    (display_multi_parent_class_icon TRUE)
    (is_readonly FALSE)
    (tabbed_instance_form_layout FALSE)
    (update_modification_slots FALSE))

([ProjectKB_0212_00022] of Property_List

    (properties
        [GeneseekDBofDBs_ProjectKB_00043]
        [GeneseekDBofDBs_ProjectKB_00044]))

([ProjectKB_0212_00034] of Property_List

    (properties [GeneseekDBofDBs_ProjectKB_00045]))

([ProjectKB_0212_00037] of Property_List

    (properties
        [GeneseekDBofDBs_ProjectKB_00046]
        [GeneseekDBofDBs_ProjectKB_00047]))

([ProjectKB_0212_00038] of Property_List

    (properties [GeneseekDBofDBs_ProjectKB_00048]))

([ProjectKB_0212_00039] of String

    (name "classes_file_name")
    (string_value "SNP.pont"))

([ProjectKB_0212_00040] of String

    (name "instances_file_name")
    (string_value "SNP.pins"))

([QUERIES_TAB] of Widget

    (is_hidden FALSE)
    (label "Queries")
    (property_list [ProjectKB_0212_00038])
    (widget_class_name
"edu.stanford.smi.protegex.queries_tab.QueriesTab"))

([SLOTS_TAB] of Widget

```

```

        (is_hidden FALSE)
        (label "Slots")
        (property_list [ProjectKB_0212_00022])
        (widget_class_name
"edu.stanford.smi.protege.widget.SlotsTab"))
([SNP_ProjectKB_Instance_20088] of String
        (name "Databases")
        (string_value "DBName"))
([SNP_ProjectKB_Instance_20089] of String
        (name "Mapping")
        (string_value "MappingName"))
([SNP_ProjectKB_Instance_20090] of String
        (name "Path")
        (string_value "PathName"))
([SNP_ProjectKB_Instance_20091] of String
        (name "MaintenanceModes")
        (string_value "ModeName"))
([SNP_ProjectKB_Instance_20092] of String
        (name ":META-CLASS")
        (string_value "%3ANAME"))
([SNP_ProjectKB_Instance_20093] of Map
        (entries
          [SNP_ProjectKB_Instance_20094]
          [SNP_ProjectKB_Instance_20095]
          [SNP_ProjectKB_Instance_20096]
          [SNP_ProjectKB_Instance_20097]))
([SNP_ProjectKB_Instance_20094] of Map_Entry
        (key "GeneseekDBofDBs_00417")
        (key_class "edu.stanford.smi.protege.model.Frame")
        (value "202 361 381 203")
        (value_class "java.awt.Rectangle"))
([SNP_ProjectKB_Instance_20095] of Map_Entry
        (key "GeneseekDBofDBs_00419")
        (key_class "edu.stanford.smi.protege.model.Frame")

```

```

    (value "25 25 213 135")
    (value_class "java.awt.Rectangle"))

([SNP_ProjectKB_Instance_20096] of Map_Entry

  (key "GeneseekDBofDBs_00418")
  (key_class "edu.stanford.smi.protege.model.Frame")
  (value "25 25 278 132")
  (value_class "java.awt.Rectangle"))

([SNP_ProjectKB_Instance_20097] of Map_Entry

  (key "GeneseekDBofDBs_00436")
  (key_class "edu.stanford.smi.protege.model.Frame")
  (value "118 160 482 313")
  (value_class "java.awt.Rectangle"))

([SNP_ProjectKB_Instance_20098] of Map_Entry

  (key "InstanceDisplay.yellow_stickies")
  (key_class "java.lang.String")
  (value "SNP_ProjectKB_Instance_20093")
  (value_class "java.util.Map"))

([SNPTest_ProjectKB_Instance_0] of Widget

  (is_hidden TRUE)
  (property_list [SNPTest_ProjectKB_Instance_1])
  (widget_class_name "TGViztab.TGVizTab"))

([SNPTest_ProjectKB_Instance_1] of Property_List
)

([SNPTest_ProjectKB_Instance_10] of Widget

  (is_hidden TRUE)
  (property_list [SNPTest_ProjectKB_Instance_11])
  (widget_class_name
"edu.stanford.smi.protege.owl.ui.metadatatab.OWLMetadataTab"))

([SNPTest_ProjectKB_Instance_10025] of Property_List
)

([SNPTest_ProjectKB_Instance_10026] of Widget

  (is_hidden TRUE)
  (property_list [SNPTest_ProjectKB_Instance_10027])
  (widget_class_name
"edu.stanford.smi.protege.prompt.PromptTab"))

([SNPTest_ProjectKB_Instance_10027] of Property_List

```

```

)

([SNPTest_ProjectKB_Instance_10028] of Widget
  (is_hidden TRUE)
  (property_list [SNPTest_ProjectKB_Instance_10029])
  (widget_class_name
"edu.stanford.smi.protege.owl.ui.properties.OWLPropertiesTab"))

([SNPTest_ProjectKB_Instance_10029] of Property_List
)

([SNPTest_ProjectKB_Instance_10030] of Widget
  (is_hidden TRUE)
  (property_list [SNPTest_ProjectKB_Instance_10031])
  (widget_class_name
"edu.stanford.smi.protege.widget.ProtegePropertiesTab"))

([SNPTest_ProjectKB_Instance_10031] of Property_List
)

([SNPTest_ProjectKB_Instance_10032] of Widget
  (is_hidden TRUE)
  (property_list [SNPTest_ProjectKB_Instance_10033])
  (widget_class_name
"edu.stanford.smi.protege.widget.pal.PalConstraintsTab"))

([SNPTest_ProjectKB_Instance_10033] of Property_List
)

([SNPTest_ProjectKB_Instance_10034] of Widget
  (is_hidden TRUE)
  (property_list [SNPTest_ProjectKB_Instance_10035])
  (widget_class_name
"edu.stanford.smi.protege.widget.instance_tree.KnowledgeTreeTab"))

([SNPTest_ProjectKB_Instance_10035] of Property_List
)

([SNPTest_ProjectKB_Instance_10036] of Widget
  (is_hidden TRUE)
  (property_list [SNPTest_ProjectKB_Instance_10037])
  (widget_class_name
"edu.stanford.smi.protege.widget.instance_tree.InstanceTreeTab"))

([SNPTest_ProjectKB_Instance_10037] of Property_List
)

```

```

([SNPTest_ProjectKB_Instance_10038] of Widget
  (is_hidden TRUE)
  (property_list [SNPTest_ProjectKB_Instance_10039])
  (widget_class_name
"edu.stanford.smi.protegex.changes.stats.ChangeStatisticsTab"))

([SNPTest_ProjectKB_Instance_10039] of Property_List
)

([SNPTest_ProjectKB_Instance_10040] of Widget
  (is_hidden TRUE)
  (property_list [SNPTest_ProjectKB_Instance_10041])
  (widget_class_name "se.liu.ida.JessTab.JessTab"))

([SNPTest_ProjectKB_Instance_10041] of Property_List
)

([SNPTest_ProjectKB_Instance_10042] of Widget
  (is_hidden TRUE)
  (property_list [SNPTest_ProjectKB_Instance_10043])
  (widget_class_name
"ca.uvic.csr.shrimp.jambalaya.JambalayaTab"))

([SNPTest_ProjectKB_Instance_10043] of Property_List
)

([SNPTest_ProjectKB_Instance_10044] of Widget
  (is_hidden TRUE)
  (property_list [SNPTest_ProjectKB_Instance_10045])
  (widget_class_name
"edu.stanford.smi.protegex.datamaster.DataMasterTab"))

([SNPTest_ProjectKB_Instance_10045] of Property_List
)

([SNPTest_ProjectKB_Instance_10046] of Widget
  (is_hidden TRUE)
  (property_list [SNPTest_ProjectKB_Instance_10047])
  (widget_class_name
"edu.stanford.smi.protegex.owl.ui.individuals.OWLIndividualsTab"))

([SNPTest_ProjectKB_Instance_10047] of Property_List
)

([SNPTest_ProjectKB_Instance_10048] of Widget

```

```

        (is_hidden TRUE)
        (property_list [SNPTest_ProjectKB_Instance_10049])
        (widget_class_name
"uk.ac.man.cs.mig.coode.debugger.test.DebuggerTestTab"))

([SNPTest_ProjectKB_Instance_10049] of Property_List
)

([SNPTest_ProjectKB_Instance_10050] of Widget

        (is_hidden TRUE)
        (property_list [SNPTest_ProjectKB_Instance_10051])
        (widget_class_name
"edu.stanford.smi.protege.owl.ui.cls.OWLClassesTab"))

([SNPTest_ProjectKB_Instance_10051] of Property_List
)

([SNPTest_ProjectKB_Instance_10052] of Widget

        (is_hidden TRUE)
        (property_list [SNPTest_ProjectKB_Instance_10053])
        (widget_class_name
"org.algernon.kb.okbc.protege.plugins.AlgernonTab"))

([SNPTest_ProjectKB_Instance_10053] of Property_List
)

([SNPTest_ProjectKB_Instance_10054] of Widget

        (is_hidden TRUE)
        (property_list [SNPTest_ProjectKB_Instance_10055])
        (widget_class_name
"edu.stanford.smi.protege.widget.pal.PalQueriesTab"))

([SNPTest_ProjectKB_Instance_10055] of Property_List
)

([SNPTest_ProjectKB_Instance_10056] of Widget

        (is_hidden TRUE)
        (property_list [SNPTest_ProjectKB_Instance_10057])
        (widget_class_name
"edu.stanford.smi.protege.chatPlugin.ChatTab"))

([SNPTest_ProjectKB_Instance_10057] of Property_List
)

([SNPTest_ProjectKB_Instance_10058] of Widget

```

```

    (is_hidden TRUE)
    (property_list [SNPTest_ProjectKB_Instance_10059])
    (widget_class_name "edu.stanford.smi.RemoteKSTab.UMLSTab"))

([SNPTest_ProjectKB_Instance_10059] of Property_List
)

([SNPTest_ProjectKB_Instance_10060] of Widget

    (is_hidden TRUE)
    (property_list [SNPTest_ProjectKB_Instance_10061])
    (widget_class_name
"edu.stanford.smi.protege.server_changes.prompt.UsersTab"))

([SNPTest_ProjectKB_Instance_10061] of Property_List
)

([SNPTest_ProjectKB_Instance_10062] of Widget

    (is_hidden TRUE)
    (property_list [SNPTest_ProjectKB_Instance_20063])
    (widget_class_name
"uk.ac.man.ac.mig.coode.individuals.ui.OwldlIndividualsTab"))

([SNPTest_ProjectKB_Instance_10063] of Widget

    (height 60)
    (is_hidden FALSE)
    (name "QsReverse")
    (property_list [SNPTest_ProjectKB_Instance_10064])
    (widget_class_name
"edu.stanford.smi.protege.widget.FloatFieldWidget")
    (width 100)
    (x 0)
    (y 480))

([SNPTest_ProjectKB_Instance_10064] of Property_List
)

([SNPTest_ProjectKB_Instance_10109] of Property_List

    (properties
      [SNPTest_ProjectKB_Instance_10110]
      [SNPTest_ProjectKB_Instance_10111]))

([SNPTest_ProjectKB_Instance_10110] of Boolean

    (boolean_value FALSE)
    (name "ButtonDisplayed-View References to Value"))

([SNPTest_ProjectKB_Instance_10111] of Boolean

```

```
(boolean_value FALSE)
(name "ButtonDisplayed-Delete Instance"))

([SNPTest_ProjectKB_Instance_10112] of Property_List
)

([SNPTest_ProjectKB_Instance_10113] of Property_List
)

([SNPTest_ProjectKB_Instance_10114] of Property_List
  (properties
    [SNPTest_ProjectKB_Instance_10115]
    [SNPTest_ProjectKB_Instance_10116]))

([SNPTest_ProjectKB_Instance_10115] of Boolean
  (boolean_value FALSE)
  (name "ButtonDisplayed-Move up"))

([SNPTest_ProjectKB_Instance_10116] of Boolean
  (boolean_value FALSE)
  (name "ButtonDisplayed-Move down"))

([SNPTest_ProjectKB_Instance_10117] of Property_List
)

([SNPTest_ProjectKB_Instance_10118] of Property_List
  (name "layout properties"))

([SNPTest_ProjectKB_Instance_10119] of Property_List
)

([SNPTest_ProjectKB_Instance_10120] of Property_List
)

([SNPTest_ProjectKB_Instance_10121] of Property_List
)

([SNPTest_ProjectKB_Instance_10122] of Property_List
)

([SNPTest_ProjectKB_Instance_10123] of Property_List
)

([SNPTest_ProjectKB_Instance_10124] of Property_List
)
```

```

([SNPTest_ProjectKB_Instance_10125] of Property_List
)

([SNPTest_ProjectKB_Instance_10126] of Property_List
)

([SNPTest_ProjectKB_Instance_10127] of Property_List
)

([SNPTest_ProjectKB_Instance_10128] of Property_List
)

([SNPTest_ProjectKB_Instance_10129] of Property_List
)

([SNPTest_ProjectKB_Instance_11] of Property_List
)

([SNPTest_ProjectKB_Instance_12] of Widget
  (is_hidden TRUE)
  (property_list [SNPTest_ProjectKB_Instance_13])
  (widget_class_name
"edu.stanford.smi.protege.changes.ChangesTab"))

([SNPTest_ProjectKB_Instance_124] of Boolean
  (boolean_value FALSE)
  (name "ButtonDisplayed-View References to Value"))

([SNPTest_ProjectKB_Instance_125] of Boolean
  (boolean_value FALSE)
  (name "ButtonDisplayed-View References to Value"))

([SNPTest_ProjectKB_Instance_126] of Boolean
  (boolean_value FALSE)
  (name "ButtonDisplayed-View References to Value"))

([SNPTest_ProjectKB_Instance_127] of Boolean
  (boolean_value FALSE)
  (name "ButtonDisplayed-View References to Value"))

([SNPTest_ProjectKB_Instance_128] of Boolean
  (boolean_value FALSE)
  (name "ButtonDisplayed-View References to Value"))

([SNPTest_ProjectKB_Instance_129] of Boolean

```

```

        (boolean_value FALSE)
        (name "ButtonDisplayed-View References to Value"))

([SNPTest_ProjectKB_Instance_13] of Property_List
)

([SNPTest_ProjectKB_Instance_14] of Widget

    (is_hidden TRUE)
    (property_list [SNPTest_ProjectKB_Instance_15])
    (widget_class_name
"edu.stanford.smi.protege.owl.ui.widget.OWLFormsTab"))

([SNPTest_ProjectKB_Instance_141] of Boolean

    (boolean_value FALSE)
    (name "ButtonDisplayed-View References to Value"))

([SNPTest_ProjectKB_Instance_15] of Property_List
)

([SNPTest_ProjectKB_Instance_16] of Widget

    (is_hidden TRUE)
    (property_list [SNPTest_ProjectKB_Instance_17])
    (widget_class_name "ezpal.EZPalTab"))

([SNPTest_ProjectKB_Instance_17] of Property_List
)

([SNPTest_ProjectKB_Instance_18] of Widget

    (is_hidden TRUE)
    (property_list [SNPTest_ProjectKB_Instance_19])
    (widget_class_name
"edu.stanford.smi.protege.owl.swrl.ui.tab.SWRLTab"))

([SNPTest_ProjectKB_Instance_19] of Property_List
)

([SNPTest_ProjectKB_Instance_2] of Widget

    (is_hidden TRUE)
    (property_list [SNPTest_ProjectKB_Instance_3])
    (widget_class_name "dfki.protege.ontoviz_tab.OntovizTab"))

([SNPTest_ProjectKB_Instance_20] of Widget

    (is_hidden TRUE)
    (property_list [SNPTest_ProjectKB_Instance_21])

```

```

        (widget_class_name
"edu.stanford.smi.protege.fctab.FacetConstraintsTab"))

([SNPTest_ProjectKB_Instance_20063] of Property_List
)

([SNPTest_ProjectKB_Instance_20064] of Widget

    (is_hidden TRUE)
    (property_list [SNPTest_ProjectKB_Instance_20065])
    (widget_class_name
"uk.ac.man.cs.mig.coode.owlviz.ui.OwlvizTab"))

([SNPTest_ProjectKB_Instance_20065] of Property_List
)

([SNPTest_ProjectKB_Instance_20154] of Boolean

    (boolean_value FALSE)
    (name "ButtonDisplayed-View References to Value"))

([SNPTest_ProjectKB_Instance_21] of Property_List
)

([SNPTest_ProjectKB_Instance_22] of Widget

    (is_hidden TRUE)
    (property_list [SNPTest_ProjectKB_Instance_23])
    (widget_class_name
"edu.stanford.smi.protege.changes.changesKBViewTab.ChangesKBViewTab
"))

([SNPTest_ProjectKB_Instance_23] of Property_List
)

([SNPTest_ProjectKB_Instance_24] of Widget

    (is_hidden TRUE)
    (property_list [SNPTest_ProjectKB_Instance_10025])
    (widget_class_name
"edu.stanford.smi.protege.xml.tab.XMLTab"))

([SNPTest_ProjectKB_Instance_25] of Property_List
)

([SNPTest_ProjectKB_Instance_26] of Property_List
)

([SNPTest_ProjectKB_Instance_27] of Property_List
)

```

```

([SNPTest_ProjectKB_Instance_3] of Property_List
)

([SNPTest_ProjectKB_Instance_4] of Widget
  (is_hidden TRUE)
  (property_list [SNPTest_ProjectKB_Instance_5])
  (widget_class_name "script.ProtegeScriptTab"))

([SNPTest_ProjectKB_Instance_5] of Property_List
)

([SNPTest_ProjectKB_Instance_6] of Widget
  (is_hidden TRUE)
  (property_list [SNPTest_ProjectKB_Instance_7])
  (widget_class_name
"edu.stanford.smi.protege.keywordsearch.StringSearch"))

([SNPTest_ProjectKB_Instance_7] of Property_List
)

([SNPTest_ProjectKB_Instance_8] of Widget
  (is_hidden TRUE)
  (property_list [SNPTest_ProjectKB_Instance_9])
  (widget_class_name "edu.stanford.smi.RemoteKCTab.WordNetTab"))

([SNPTest_ProjectKB_Instance_9] of Property_List
)

([SOURCES] of Property_List
  (properties
    [ProjectKB_0212_00000]
    [ProjectKB_0212_00039]
    [ProjectKB_0212_00040]))

([STANDARD_CLASS_FORM_WIDGET] of Widget
  (height 597)
  (name ":STANDARD-CLASS")
  (property_list [KB_862621_Instance_0])
  (widget_class_name
"edu.stanford.smi.protege.widget.FormWidget")
  (width 738)
  (x 0)
  (y 0))

([STANDARD_SLOT_FORM_WIDGET] of Widget

```

```
(height 597)
(name ":STANDARD-SLOT")
(property_list [KB_862621_Instance_12])
(widget_class_name
"edu.stanford.smi.protege.widget.FormWidget")
(width 738)
(x 0)
(y 0))
```

Appendix C: Belief Resolver java files

```
/**
 *
 */
package org.biomediator.belief.datasource;

import org.biomediator.belief.BeliefResolverStub;
import org.biomediator.belief.exception.InvalidParameterException;
import org.biomediator.belief.exception.SKBException;
import org.biomediator.belief.struct.ResourceU2;
import org.biomediator.belief.struct.U2;
import org.biomediator.belief.utils.scale.NonLinearScale;
import org.jdom.Element;

import edu.stanford.smi.protege.model.KnowledgeBase;
import edu.washington.cs.db.browser.biomediator.BioMediatorSAXEntity;
import
    edu.washington.cs.db.browser.biomediator.BioMediatorResourceTable.BioM
    ediatorResource;

import java.util.*;

/**
 * @author detwiler
 * @date Jun 9, 2006
 */
public class dbSNP_BeliefResolver extends BeliefResolverStub
{

    // vals for SNP scale
    private final double HAPMAP = 1.0;
    private final double FREQUENCY = 0.9;
    private final double TWO_CHROMOSOME = 0.8;
    private final double SUBMITTER_CONFIRMED = 0.6;
    private final double MULTIPLE = 0.6;

    private final String ValidationMethod = "//ValidationMethod";

    private NonLinearScale snpScale = new NonLinearScale();
```

```

/**
 * @param kb
 * @throws SKBException
 */
public dbSNP_BeliefResolver(KnowledgeBase kb) throws SKBException
//someone calls this
{
    super(kb); //gets all the Qs, Ps, stuff from knowledge base
    init();
}

private boolean init() //initialization method sets up all initial data structures,
anything you need later
{
    try
    {
        snpScale.addElement("by-hapmap", HAPMAP);
        snpScale.addElement("by-frequency", FREQUENCY);
        snpScale.addElement("by-2hit-2allele",
TWO_CHROMOSOME);
        snpScale.addElement("by-submitter",
SUBMITTER_CONFIRMED);
        snpScale.addElement("by-cluster", MULTIPLE);

    }
    catch (InvalidParameterException e)
    {
        e.printStackTrace();
        return false;
    }

    return true;
}

/* (non-Javadoc)

```

```

* @see
org.biomediator.belief.BeliefResolverStub#Pr(org.biomediator.belief.struct.R
esourceU2,
edu.washington.cs.db.browser.biomediator.BioMediatorResourceTable.BioM
ediatorResource)
*/
@Override
public float Pr(ResourceU2 u2, BioMediatorResource resource) throws
InvalidParameterException //someone calls this
{
    if("SNP".equals(resource.getTypeName()))
    {
        return snpPr(resource);
    }
    else
    {
        return getDefaultPr();
    }
}

public float snpPr(BioMediatorResource resource) throws
InvalidParameterException
{
    BioMediatorSAXEntity info = resource.saxEntity;
    if(info==null)
        return getDefaultPr();

    Element statusElement = null;
    statusElement = info.getElement(ValidationMethod);

    if(statusElement!=null)
    {
        // get the scale key
        String status = statusElement.getText();

        //insert code here that parses that that I can get all the values

        //System.out.println("status text: " + status);

        List<String> validationCodes = new ArrayList<String>();

```

```

StringTokenizer st;
st = new StringTokenizer (status,",");

while (st.hasMoreTokens()) {
    String myNextElement = st.nextToken();
    validationCodes.add(myNextElement);
}

float pr = 0.0f;

Iterator validCodesIt = validationCodes.iterator();
while (validCodesIt.hasNext()){
    String currCode = (String)validCodesIt.next();
    float currScore = snpScale.getScaleValue(currCode);

    if (currScore > pr)
        pr = currScore;
}

// get the scale value associated with this scale key
//float pr = snpScale.getScaleValue(status);

return pr;
}
else
{
    //debug
    System.err.println("Warning, default Pr value returned for
dbSNP_BeliefResolver.Pr()");

    return getDefaultPr();
}
}

/**
 *
 */

```

```

package org.biomediator.belief.datasource;

import org.biomediator.belief.BeliefResolverStub;
import org.biomediator.belief.exception.InvalidParameterException;
import org.biomediator.belief.exception.SKBException;
import org.biomediator.belief.struct.ResourceU2;
import org.biomediator.belief.struct.U2;
import org.jdom.Element;

import edu.stanford.smi.protege.model.KnowledgeBase;
import edu.washington.cs.db.browser.biomediator.BioMediatorSAXEntity;
import
    edu.washington.cs.db.browser.biomediator.BioMediatorResourceTable.BioM
    ediatorResource;

/**
 * @author terry shen
 * @date July 15, 2008
 */
public class GVS_BeliefResolver extends BeliefResolverStub
{

    private final String ValidationMethod = "//r2Score";

    /**
     * @param kb
     * @throws SKBException
     */
    public GVS_BeliefResolver(KnowledgeBase kb) throws SKBException
    {
        super(kb);
        init();
    }

    private boolean init()
    {
        return true;
    }

    /* (non-Javadoc)

```

```

* @see
org.biomediator.belief.BeliefResolverStub#Pr(org.biomediator.belief.struct.U
2, java.lang.String)
*/
/*
@Override
public float Pr(U2 u2, String results) throws InvalidParameterException
{
    return getDefaultPr();
}
*/

/* (non-Javadoc)
* @see
org.biomediator.belief.BeliefResolverStub#Pr(org.biomediator.belief.struct.R
esourceU2,
edu.washington.cs.db.browser.biomediator.BioMediatorResourceTable.BioM
ediatorResource)
*/
@Override
public float Pr(ResourceU2 u2, BioMediatorResource resource) throws
InvalidParameterException
{
    if("LinkageDisequilibriumPopulation".equals(resource.getTypeName(
)))
    {
        return LDPr(resource);
    }
    else
    {
        return getDefaultPr();
    }
}

public float LDPr(BioMediatorResource resource) throws
InvalidParameterException
{
    BioMediatorSAXEntity info = resource.saxEntity;
    if(info==null)

```

```

        return getDefaultPr();

        Element statusElement = null;
        statusElement = info.getElement(ValidationMethod);

        if(statusElement!=null)
        {
            // get the scale key
            String status = statusElement.getText();

            //insert code here that parses that that I can get all the values
            //
            //System.out.println("THIS IS THE LINKAGE
DISEQUILIBRIUM SCORE: " + status);
            float pr = Float.parseFloat(status);
            float newPr = (pr*0.5f)+0.5f;
            //System.err.println("THIS IS THE REVISED PR SCORE
FOR LD: " + newPr);
            return newPr;

        }
        else
        {
            //debug
            System.err.println("Warning, default Pr value returned for
GVS_BeliefResolver.Pr()");

            return getDefaultPr();
        }
    }

    /* (non-Javadoc)
    * @see org.biomediator.belief.BeliefResolverStub#getDefaultPr()
    */
    @Override
    protected float getDefaultPr()
    {
        return 1.0f;
    }
}

```

```

/**
 *
 */
package org.biomediator.belief.datasource;

import org.biomediator.belief.BeliefResolverStub;
import org.biomediator.belief.exception.InvalidParameterException;
import org.biomediator.belief.exception.SKBException;
import org.biomediator.belief.struct.ResourceU2;
import org.biomediator.belief.struct.U2;

import edu.stanford.smi.protege.model.KnowledgeBase;
import
    edu.washington.cs.db.browser.biomediator.BioMediatorResourceTable.BioM
    ediatorResource;

/**
 * @author detwiler
 * @date Jun 9, 2006
 */
public class Haplotter_BeliefResolver extends BeliefResolverStub
{
    /**
     * @param kb
     * @throws SKBException
     */
    public Haplotter_BeliefResolver(KnowledgeBase kb) throws SKBException
    {
        super(kb);
        init();
    }

    private boolean init()
    {
        return true;
    }

    /* (non-Javadoc)

```

```

    * @see
    org.biomediator.belief.BeliefResolverStub#Pr(org.biomediator.belief.struct.U
    2, java.lang.String)
    */
    /*
    @Override
    public float Pr(U2 u2, String results) throws InvalidParameterException
    {
        return getDefaultPr();
    }
    */

    /* (non-Javadoc)
    * @see
    org.biomediator.belief.BeliefResolverStub#Pr(org.biomediator.belief.struct.R
    esourceU2,
    edu.washington.cs.db.browser.biomediator.BioMediatorResourceTable.BioM
    ediatorResource)
    */
    @Override
    public float Pr(ResourceU2 u2, BioMediatorResource resource) throws
    InvalidParameterException
    {
        return getDefaultPr();
    }

    /* (non-Javadoc)
    * @see org.biomediator.belief.BeliefResolverStub#getDefaultPr()
    */
    @Override
    protected float getDefaultPr()
    {
        return 0.5f;
    }
}

/**
 *
 */
package org.biomediator.belief.datasource;

```

```

import org.biomediator.belief.BeliefResolverStub;
import org.biomediator.belief.exception.InvalidParameterException;
import org.biomediator.belief.exception.SKBException;
import org.biomediator.belief.struct.ResourceU2;
import org.biomediator.belief.struct.U2;

import edu.stanford.smi.protege.model.KnowledgeBase;
import
    edu.washington.cs.db.browser.biomediator.BioMediatorResourceTable.BioM
    ediatorResource;

/**
 * @author detwiler
 * @date Jun 9, 2006
 */
public class HGMD_BeliefResolver extends BeliefResolverStub
{
    /**
     * @param kb
     * @throws SKBException
     */
    public HGMD_BeliefResolver(KnowledgeBase kb) throws SKBException
    {
        super(kb);
        init();
    }

    private boolean init()
    {
        return true;
    }

    /* (non-Javadoc)
     * @see
     org.biomediator.belief.BeliefResolverStub#Pr(org.biomediator.belief.struct.U
     2, java.lang.String)
     */
    /*
     @Override
     public float Pr(U2 u2, String results) throws InvalidParameterException

```

```

    {
        return getDefaultPr();
    }
    */

    /* (non-Javadoc)
     * @see
     org.biomediator.belief.BeliefResolverStub#Pr(org.biomediator.belief.struct.ResourceU2,
     edu.washington.cs.db.browser.biomediator.BioMediatorResourceTable.BioMediatorResource)
     */
    @Override
    public float Pr(ResourceU2 u2, BioMediatorResource resource) throws
    InvalidParameterException
    {
        return getDefaultPr();
    }

    /* (non-Javadoc)
     * @see org.biomediator.belief.BeliefResolverStub#getDefaultPr()
     */
    @Override
    protected float getDefaultPr()
    {
        return 0.5f;
    }
}

/**
 *
 */
package org.biomediator.belief.datasource;

import org.biomediator.belief.BeliefResolverStub;
import org.biomediator.belief.exception.InvalidParameterException;
import org.biomediator.belief.exception.SKBException;
import org.biomediator.belief.struct.ResourceU2;
import org.jdom.Element;

```

```

import edu.stanford.smi.protege.model.KnowledgeBase;
import edu.washington.cs.db.browser.biomediator.BioMediatorSAXEntity;
import
    edu.washington.cs.db.browser.biomediator.BioMediatorResourceTable.BioM
    ediatorResource;

/**
 * @author terry shen
 *
 */
public class SIFT_BeliefResolver extends BeliefResolverStub
{

    private final String ValidationMethod1 = "//Score1Homologue";
    private final String ValidationMethod2 = "//Score2Homologue";

    private float pr = 0.0f;

    /**
     * @param kb
     * @throws SKBException
     */
    public SIFT_BeliefResolver(KnowledgeBase kb) throws SKBException
    {
        super(kb);
        init();
    }

    private boolean init()
    {
        return true;
    }

    /* (non-Javadoc)
     * @see
     org.biomediator.belief.BeliefResolverStub#Pr(org.biomediator.belief.struct.R
     ourceU2,
     edu.washington.cs.db.browser.biomediator.BioMediatorResourceTable.BioM
     ediatorResource)
     */
    @Override

```

```

public float Pr(ResourceU2 u2, BioMediatorResource resource) throws
InvalidParameterException
{
    if("ProteinFunctionPrediction".equals(resource.getTypeName()))
    {
        return proteinFunctionPredictionPr(resource);
    }
    else
    {
        return getDefaultPr();
    }
}

```

```

public float proteinFunctionPredictionPr(BioMediatorResource resource)
throws InvalidParameterException
{
    BioMediatorSAXEntity info = resource.saxEntity;
    if(info==null)
        return getDefaultPr();

    Element statusElement1 = info.getElement(ValidationMethod1);
    Element statusElement2 = info.getElement(ValidationMethod2);

    String stringElement1 = statusElement1.getText();
    String stringElement2 = statusElement2.getText();

    System.out.println("THIS SNP HAS A SIFT PAIR1 PR SCORE: " +
stringElement1);
    System.out.println("THIS SNP HAS A SIFT PAIR2 PR SCORE: " +
stringElement2);

    float floatElement1 = Float.valueOf(stringElement1);
    float floatElement2 = Float.valueOf(stringElement2);

    if(floatElement1 > floatElement2)
    {
        //need to add on top of .5, so it's (the number)*.5, then add that
to .5
        if (floatElement1 >= 0.05 && floatElement1 < 0.1){
            pr = 1.0f;

```

```
    } else if (floatElement1 >= 0.1 && floatElement1 < 0.15){
        pr = 0.9f;
    } else if (floatElement1 >= 0.15 && floatElement1 < 0.2){
        pr = 0.85f;
    } else if (floatElement1 >= 0.2 && floatElement1 < 0.25){
        pr = 0.8f;
    } else if (floatElement1 >= 0.25 && floatElement1 < 0.3){
        pr = 0.75f;
    } else if (floatElement1 >= 0.3 && floatElement1 < 0.35){
        pr = 0.725f;
    } else if (floatElement1 >= 0.35 && floatElement1 < 0.4){
        pr = 0.7f;
    } else if (floatElement1 >= 0.4 && floatElement1 < 0.45){
        pr = 0.675f;
    } else if (floatElement1 >= 0.45 && floatElement1 < 0.5){
        pr = 0.66f;
    } else if (floatElement1 >= 0.5 && floatElement1 < 0.55){
        pr = 0.65f;
    } else if (floatElement1 >= 0.55 && floatElement1 < 0.6){
        pr = 0.625f;
    } else if (floatElement1 >= 0.6 && floatElement1 < 0.65){
        pr = 0.61f;
    } else if (floatElement1 >= 0.65 && floatElement1 < 0.7){
        pr = 0.6f;
    } else if (floatElement1 >= 0.7 && floatElement1 < 0.75){
        pr = 0.575f;
    } else if (floatElement1 >= 0.75 && floatElement1 < 0.8){
        pr = 0.56f;
    } else if (floatElement1 >= 0.8 && floatElement1 < 0.85){
        pr = 0.555f;
    } else {
        pr = 0.55f;
    }

    return pr;
}
```

```
else if(floatElement2 > floatElement1)
{
```

to .5

//need to add on top of .5, so it's (the number)*.5, then add that

```

if (floatElement2 >= 0.05 && floatElement2 < 0.1){
    pr = 1.0f;
} else if (floatElement2 >= 0.1 && floatElement2 < 0.15){
    pr = 0.9f;
} else if (floatElement2 >= 0.15 && floatElement2 < 0.2){
    pr = 0.85f;
} else if (floatElement2 >= 0.2 && floatElement2 < 0.25){
    pr = 0.8f;
} else if (floatElement2 >= 0.25 && floatElement2 < 0.3){
    pr = 0.75f;
} else if (floatElement2 >= 0.3 && floatElement2 < 0.35){
    pr = 0.725f;
} else if (floatElement2 >= 0.35 && floatElement2 < 0.4){
    pr = 0.7f;
} else if (floatElement2 >= 0.4 && floatElement2 < 0.45){
    pr = 0.675f;
} else if (floatElement2 >= 0.45 && floatElement2 < 0.5){
    pr = 0.66f;
} else if (floatElement2 >= 0.5 && floatElement2 < 0.55){
    pr = 0.65f;
} else if (floatElement2 >= 0.55 && floatElement2 < 0.6){
    pr = 0.625f;
} else if (floatElement2 >= 0.6 && floatElement2 < 0.65){
    pr = 0.61f;
} else if (floatElement2 >= 0.65 && floatElement2 < 0.7){
    pr = 0.6f;
} else if (floatElement2 >= 0.7 && floatElement2 < 0.75){
    pr = 0.575f;
} else if (floatElement2 >= 0.75 && floatElement2 < 0.8){
    pr = 0.56f;
} else if (floatElement2 >= 0.8 && floatElement2 < 0.85){
    pr = 0.555f;
} else {
    pr = 0.55f;
}

return pr;
}

```

```

        else
        {
            //debug
            System.err.println("Warning, default Pr value returned for
SIFT_BeliefResolver.Pr()");

            return getDefaultPr();
        }
    }

    @Override
    protected float getDefaultPr()
    {
        return 0.5f;
    }
}

/**
 *
 */
package org.biomediator.belief.datasource;

import org.biomediator.belief.BeliefResolverStub;
import org.biomediator.belief.exception.InvalidParameterException;
import org.biomediator.belief.exception.SKBException;
import org.biomediator.belief.struct.ResourceU2;
import org.biomediator.belief.struct.U2;
import org.jdom.Element;

import edu.stanford.smi.protege.model.KnowledgeBase;
import edu.washington.cs.db.browser.biomediator.BioMediatorSAXEntity;
import
    edu.washington.cs.db.browser.biomediator.BioMediatorResourceTable.BioM
ediatorResource;

/**
 * @author terry shen

```

```

* @date Jan 10, 2009
*/
public class TRANSFAC_BeliefResolver extends BeliefResolverStub
{

    //private final String ValidationMethod = "//TFScore";

    /**
    * @param kb
    * @throws SKBException
    */
    public TRANSFAC_BeliefResolver(KnowledgeBase kb) throws
    SKBException
    {
        super(kb);
        init();
    }

    private boolean init()
    {
        return true;
    }

    /* (non-Javadoc)
    * @see
    org.biomediator.belief.BeliefResolverStub#Pr(org.biomediator.belief.struct.U
    2, java.lang.String)
    */
    /*
    @Override
    public float Pr(U2 u2, String results) throws InvalidParameterException
    {
        return getDefaultPr();
    }
    */

    /* (non-Javadoc)
    * @see
    org.biomediator.belief.BeliefResolverStub#Pr(org.biomediator.belief.struct.R
    esourceU2,

```

```

edu.washington.cs.db.browser.biomediator.BioMediatorResourceTable.BioM
ediatorResource)
*/
@Override
public float Pr(ResourceU2 u2, BioMediatorResource resource) throws
InvalidParameterException
{
    if("TranscriptionFactorBinding".equals(resource.getTypeName()))
    {
        //return tfRecordPr(resource);
        return 0.542f;
    }
    else
    {
        return getDefaultPr();
    }
}

```

```

public float tfRecordPr(BioMediatorResource resource) throws
InvalidParameterException
{
    BioMediatorSAXEntity info = resource.saxEntity;
    if(info==null)
        return getDefaultPr();

    Element statusElement = null;
    //statusElement = info.getElement(ValidationMethod);

    if(statusElement!=null)
    {
        // get the scale key
        String status = statusElement.getText();

        //insert code here that parses that that I can get all the values

        System.out.println("THIS SNP HAS A TRANSFAC PR
SCORE: " + status);
    }
}

```

```

        float pr = 0.0f;
        pr = Float.parseFloat(status);

        return 0.542f;
    }
    else
    {
        //debug
        System.err.println("Warning, default Pr value returned for
BDGP_BeliefResolver.Pr()");

        return getDefaultPr();
    }
}

/* (non-Javadoc)
 * @see org.biomediator.belief.BeliefResolverStub#getDefaultPr()
 */
@Override
protected float getDefaultPr()
{
    return 0.5f;
}
}

/**
 *
 */
package org.biomediator.belief.datasource;

import org.biomediator.belief.BeliefResolverStub;
import org.biomediator.belief.exception.InvalidParameterException;
import org.biomediator.belief.exception.SKBException;
import org.biomediator.belief.struct.ResourceU2;
import org.biomediator.belief.struct.U2;

```

```

import edu.stanford.smi.protege.model.KnowledgeBase;
import
    edu.washington.cs.db.browser.biomediator.BioMediatorResourceTable.BioM
    ediatorResource;

/**
 * @author detwiler
 * @date Jun 9, 2006
 */
public class UCSC_BeliefResolver extends BeliefResolverStub
{
    /**
     * @param kb
     * @throws SKBException
     */
    public UCSC_BeliefResolver(KnowledgeBase kb) throws SKBException
    {
        super(kb);
        init();
    }

    private boolean init()
    {
        return true;
    }

    /* (non-Javadoc)
     * @see
     org.biomediator.belief.BeliefResolverStub#Pr(org.biomediator.belief.struct.U
     2, java.lang.String)
     */
    /*
     @Override
     public float Pr(U2 u2, String results) throws InvalidParameterException
     {
         return getDefaultPr();
     }
     */

    /* (non-Javadoc)

```

```
* @see
org.biomediator.belief.BeliefResolverStub#Pr(org.biomediator.belief.struct.R
resourceU2,
edu.washington.cs.db.browser.biomediator.BioMediatorResourceTable.BioM
ediatorResource)
*/
@Override
public float Pr(ResourceU2 u2, BioMediatorResource resource) throws
InvalidParameterException
{
    return getDefaultPr();
}

/* (non-Javadoc)
 * @see org.biomediator.belief.BeliefResolverStub#getDefaultPr()
 */
@Override
protected float getDefaultPr()
{
    return 0.5f;
}
}
```



```

;rule that check cSNPs that are nonsynonymous
(defrule check_cSNP_nonsyn
  (SNP (SourceID ?snp_id) (PredictedFunctionalRole ?a&:(regex ?a
"missense,reference")))
  (not(ProteinFunctionPrediction (SourceID ?snp_id)
(Prediction1Homologue ?b)))
  =>
  (printout t "cSNP nonsyn" crlf)
  (assert (RankingSNP (rsnumber ?snp_id) (score (format nil "%.3f"
2.25)) (category "coding SNP, nonsynonymous")))
)

```

```

;rule that check cSNPs that are nonsynonymous and SIFT damaging
(defrule check_cSNP_nonsyn_damaging
  (SNP (SourceID ?snp_id) (PredictedFunctionalRole ?a&:(regex ?a
"missense,reference")))
  (ProteinFunctionPrediction (SourceID ?snp_id)
(Prediction1Homologue ?b&:(eq ?b "DAMAGING")))
  =>
  (assert (RankingSNP (rsnumber ?snp_id) (score (format nil "%.3f"
3.375)) (category "coding SNP, nonsynonymous, damaging")))
)

```

```

;rule that check cSNPs that are nonsynonymous and SIFT benign
(defrule check_cSNP_nonsyn_tolerated
  (SNP (SourceID ?snp_id) (PredictedFunctionalRole ?a&:(regex ?a
"missense,reference")))
  (ProteinFunctionPrediction (SourceID ?snp_id)
(Prediction1Homologue ?b&:(eq ?b "TOLERATED")))
  =>
  (assert (RankingSNP (rsnumber ?snp_id) (score (format nil "%.3f"
2.25)) (category "coding SNP, nonsynonymous, benign")))
)

```

```

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;

```

```

;rule that check cSNPs that are synonymous
(defrule check_cSNP_syn
  (SNP (SourceID ?snp_id) (PredictedFunctionalRole ?a&:(regex ?a
"coding-synonymous")))
  (not(ProteinFunctionPrediction (SourceID ?snp_id)
(Prediction1Homologue ?b)))
  =>

```

```

(printout t "cSNP nonsyn" crlf)
(assert (RankingSNP (rsnumber ?snp_id) (score (format nil "%.3f"
2.25)) (category "coding SNP, synonymous")))
)

```

```

;rule that check cSNPs that are synonymous and high ECR
(defrule check_cSNP_syn_highECR
  (SNP (SourceID ?snp_id) (PredictedFunctionalRole ?a&:(regex ?a
"coding-synonymous")) (ChromosomeStart ?chrom_loc)
  (EvolutionaryConservedRegions (SourceID ?id) (PhastconScore
?ecr_score&:(> ?ecr_score 0.5)) (Location ?chrom_loc))
  =>
  (assert (RankingSNP (rsnumber ?snp_id) (score (format nil "%.3f"
(* 0.75 (+ 0.5 ?ecr_score)))) (category "coding SNP, synonymous,
high evolutionary conservation")))
)

```

```

;rule that check cSNPs that are synonymous and low ECR
(defrule check_cSNP_syn_lowECR
  (SNP (SourceID ?snp_id) (PredictedFunctionalRole ?a&:(regex ?a
"coding-synonymous")) (ChromosomeStart ?chrom_loc)
  (EvolutionaryConservedRegions (SourceID ?id) (PhastconScore
?ecr_score&:(< ?ecr_score 0.5)) (Location ?chrom_loc))
  =>
  (assert (RankingSNP (rsnumber ?snp_id) (score (format nil "%.3f"
(* 0.75 (+ 0.5 ?ecr_score)))) (category "coding SNP, synonymous, low
evolutionary conservation")))
)

```

```

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;

```

```

;rule that check non-coding SNPs that are in the UTR
(defrule check_noncodingSNP_3UTR
  (SNP (SourceID ?snp_id) (PredictedFunctionalRole ?a&:(regex ?a
"utr-3")))
  =>
  (assert (RankingSNP (rsnumber ?snp_id) (score (format nil "%.3f"
1.5)) (category "non-coding SNP, 3' UTR")))
)

```

```

(defrule check_noncodingSNP_5UTR
  (SNP (SourceID ?snp_id) (PredictedFunctionalRole ?a&:(regex ?a
"utr-5")))
  =>

```

```

    (assert (RankingSNP (rsnumber ?snp_id) (score (format nil "%.3f"
2.25)) (category "non-coding SNP, 5' UTR")))
)

```

```

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;

```

```

;rule that check SNPs in the intron
(defrule check_intronicSNP
  (SNP (SourceID ?snp_id) (PredictedFunctionalRole ?a&:(regex ?a
"intron")) (ChromosomeStart ?chrom_loc))
  (not (EvolutionaryConservedRegions (SourceID ?id) (PhastconScore
?ecr_score) (Location ?chrom_loc)))
  =>
  (printout t "intron SNP" crlf)
  (assert (RankingSNP (rsnumber ?snp_id) (score (format nil "%.3f"
1.2)) (category "intronic SNP")))
)

```

```

;rule that check SNPs in the intron and high ECR
(defrule check_intronicSNP_highECR
  (SNP (SourceID ?snp_id) (PredictedFunctionalRole ?a&:(regex ?a
"intron")) (ChromosomeStart ?chrom_loc))
  (EvolutionaryConservedRegions (SourceID ?id) (PhastconScore
?ecr_score&:(> ?ecr_score 0.5)) (Location ?chrom_loc))
  =>
  (printout t "intron SNP, high ECR" crlf)
  (assert (RankingSNP (rsnumber ?snp_id) (score (format nil "%.3f"
(* 1.20 (+ 0.5 ?ecr_score))))) (category "intronic, high evolutionary
conservation")))
)

```

```

;rule that check SNPs in the intron and low ECR
(defrule check_intronicSNP_lowECR
  (SNP (SourceID ?snp_id) (PredictedFunctionalRole ?a&:(regex ?a
"intron")) (ChromosomeStart ?chrom_loc))
  (EvolutionaryConservedRegions (SourceID ?id) (PhastconScore
?ecr_score&:(< ?ecr_score 0.5)) (Location ?chrom_loc))
  =>
  (printout t "intron SNP, low ECR" crlf)
  (assert (RankingSNP (rsnumber ?snp_id) (score (format nil "%.3f"
(* 1.20 (+ 0.5 ?ecr_score))))) (category "intronic, low evolutionary
conservation")))
)

```

```
////////////////////////////////////  
////////////////////////////////////  
////////////////////////////////////
```

```
;rule that check cSNPs that are non-coding, non-UTR, flanking  
(defrule check_noncSNP_nonUTR_flanking  
  (SNP (SourceID ?snp_id) (PredictedFunctionalRole ?a&:(regex ?a  
"near-gene-3")))  
  =>  
  (assert (RankingSNP (rsnumber ?snp_id) (score (format nil "%.3f"  
1.0)) (category "non-coding SNP, non-UTR, flanking SNP, near-gene-  
3")))  
)
```

```
(defrule check_noncSNP_nonUTR_flanking  
  (SNP (SourceID ?snp_id) (PredictedFunctionalRole ?a&:(regex ?a  
"near-gene-5")))  
  =>  
  (assert (RankingSNP (rsnumber ?snp_id) (score (format nil "%.3f"  
1.0)) (category "non-coding SNP, non-UTR, flanking SNP, near-gene-  
5")))  
)
```

```
////////////////////////////////////  
////////////////////////////////////  
////////////////////////////////////
```

Appendix E: Qualitative study consent form and questions

UNIVERSITY OF WASHINGTON CONSENT FORM

Formative Interviews for the Development of an Instrument to Assess SNPit

Researchers: Terry Shen, Graduate Student, (206) 321-2003
Peter Tarczy-Hornoch, Professor, Division Head

Researchers' statement

We are asking for you to participate in a research study. The purpose of this consent form is to provide you with the information needed to decide whether you would agree to be part of this study or not. You are encouraged to ask questions about the purpose of this research, what role we ask that you play in the study, any possibly risks and benefits of the study, and any other areas of this form that is not clear concerning this study. When all your questions have been answered, you can decide whether or not you would like to participate in this study. This process is called "informed consent". You may save a copy of this form for your records.

PURPOSE OF THE STUDY

The purpose of this research is to create a survey that can measure the utility of a SNP Integration Tool entitled SNPit. Defined as both usability and usefulness, the measurement of utility will help enable us to improve the SNPit tool. The interviews conducted in this study will help in the formation of a survey to measure utility.

STUDY PROCEDURES

This study involves spending 60 minutes with each of the participants conducting semi-structured interviews to develop a utility survey for the final SNPit tool. The participants will be instructed concerning the general goals of this project and given the chance to navigate through the web servlet. The interviews will then consist of

two SNP annotation tasks to complete using the SNPit tool asking the participants to do a “think-aloud” as they use the tool focusing on both usability and usefulness. The second part of the interviews will consist of three sets of open ended questions aimed at eliciting what questions would be useful for a survey to assess a) usability, and b) usefulness.

RISKS, STRESS, OR DISCOMFORT

If you have any questions or experience any risk, stress or discomfort completing the surveys, please contact one of the investigators listed above.

Initial Open-ended Questions for Second Phase of Interview

1. How usable is the SNPit system in your opinion? Is the navigation easy to understand? Do you have a difficult time reading the results? What areas of the web site would you change?
2. How useful is the SNPit system for fellow researchers in your opinion? What kinds of scenarios do you envision such a tool being used? How might we improve the usefulness of the SNPit system?
3. What kinds of questions would you like to see on the online survey concerning SNPit? Would you prefer a Likert scale questionnaire, open-ended questions, or a combination of both?

VITA

Terry Hsin-Yi Shen received her bachelor's degree in Information Technology, with a minor in Biology, from the College of William and Mary. She completed her MSPH in Public Health Informatics from Emory University and her PhD in Biomedical Informatics from the University of Washington. Her previous work dealt with combining epidemiologic and genetic data, which was the subject of her master's thesis. Her PhD dissertation dealt with the creation of an analytical system for the purposes of functional SNP annotation. She has collaborated with the CDC, Emory's Public Health Preparedness department, as well as the Fred Hutchinson Cancer Research Center on various projects. Terry's current research interest continues to be exploring new intersections between Public Health Informatics and Bioinformatics.