

**SIDEKICK: INTEGRATING KNOWLEDGE AND USER BELIEF
TO ENABLE BIOLOGICAL DISCOVERY**

APPROVED BY SUPERVISING COMMITTEE:

Kay A. Robbins, Ph.D., Chair

Tom Bylander, Ph.D.

Jianhua Ruan, Ph.D.

Dakai Zhu, Ph.D.

Yufei Huang, Ph.D.

Accepted: _____
Dean, Graduate School

Copyright 2010 Mark S. Doderer
All Rights Reserved

DEDICATION

This work is dedicated to my family; my parents who always have believed in me, my children who inspire me, and my wife who makes my life wonderful and complete.

**SIDEKICK: INTEGRATING KNOWLEDGE AND USER BELIEF
TO ENABLE BIOLOGICAL DISCOVERY**

by

MARK S. DODERER, M.S.

DISSERTATION
Presented to the Graduate Faculty of
The University of Texas at San Antonio
In Partial Fulfillment
Of the Requirements
For the Degree of

DOCTOR OF PHILOSOPHY IN COMPUTER SCIENCE

THE UNIVERSITY OF TEXAS AT SAN ANTONIO
College of Sciences
Department of Computer Science
August 2010

ACKNOWLEDGEMENTS

I am indebted to Dr. Kay A. Robbins, my advisor. Her pursuit of excellence in scholarship combined with her depth of knowledge and experience have been a driving force in the creation of Sidekick. Beyond academic work she has been an understanding and caring mentor focused on taking care of one's family especially when there are special needs.

I am also grateful to Dr. Kihoon Yoon. When we were fellow students, Kihoon converted several students, including me from theoretical machine learning to bioinformatics. Leading us through the fundamentals of molecular biology and how computation can benefit this field, Kihoon enabled and continues to influence my work.

I want to thank Cory Burkhardt for his initial work on SideCache and to acknowledge grant support from the NIH Research Centers in Minority Institutions 2G12RR1364-06A1.

August 2010

SIDEKICK: INTEGRATING KNOWLEDGE AND USER BELIEF TO ENABLE BIOLOGICAL DISCOVERY

Mark S. Doderer, Ph.D.
The University of Texas at San Antonio, 2010

Supervising Professor: Kay A. Robbins, Ph.D.

Scientists striving to unlock mysteries within complex biological systems face myriad barriers in effectively integrating available information to enhance their understanding. Useful information is dispersed across a variety of sources, and sources of the same information often do not use the same format or nomenclature. Scientists need tools that bridge nomenclature differences and allow them to integrate, organize, and evaluate the quality of information without extensive computation.

Sidekick, a genomic data-driven analysis and decision-making application, is a web-based collection of tools to query for information, combine, manage and evaluate the information and discover connections among the information. Unlike a workflow with a series of prescribed steps, the application of tools provides a flexible solution to the problem of knowledge discovery. Sidekick enables scientists without training in computation and data management to pursue research questions like “Are there other mechanisms for disease X” or “Does the set of genes associated with disease X also influence other diseases” without prescribing the steps to answer the questions. Sidekick enables the process of combining heterogeneous data, finding and maintaining the most up-to-date data, evaluating data sources, quantifying confidence in results based on evidence, and managing the multi-step research tasks needed to answer these questions. Possible analysis steps include gene list discovery, gene-pair list discovery, and enrichment for both types of lists.

Similar tools exist for some of these tasks, however Sidekick’s Belief Manager provides a new visualization approach for quantifying confidence in results based on evidence and user belief. Also, Sidekick's ability to characterize pairs of genes offers a new way to understand gene relationships that traditional single gene lists do not, particularly in areas such as interaction

discovery. We demonstrate Sidekick's use of these new tools and overall effectiveness and flexibility by showing how to accomplish a complex published analysis involving protein-protein interactions in a fraction of the original time; while encapsulating the computational work and hiding it from the Sidekick user.

TABLE OF CONTENTS

Acknowledgements.....	iv
Abstract.....	v
List of Tables	xi
List of Figures.....	xii
CHAPTER ONE: THE NATURE OF BIOLOGICAL KNOWLEDGE DISCOVERY	1
1.1 Introduction.....	1
1.2 The bioinformatics research process.....	3
1.2.1 Manual database search	3
1.2.2 Research by using tools	4
1.3 A comparison of Sidekick, InterologueFinder, Cytoscape and DAVID	6
1.3.1 Transparent integration of information	8
1.3.2 Belief management	9
1.3.3 Pair-wise analysis.....	12
1.3.4 Workspace and analysis management	13
1.3.5 Automatic updating of information	13
1.3.6 Extensibility	15
1.4 Computational overview.....	15
1.5 Overview of this thesis.....	20
CHAPTER TWO: GENES, PROTEIN INTERACTIONS AND DISEASE	22
2.1 Biology of protein-protein interactions.....	22
2.2 Experimental methods for determining interactions.....	25
2.2.1 Yeast two-hybrid.....	25
2.2.2 Mass spectroscopy	26
2.2.3 Nuclear magnetic resonance spectroscopy	26
2.2.4 Co-immunoprecipitation	26

2.2.5 Gene expression	26
2.2.6 X-Ray crystallography	27
2.3 Computational methods for determining interactions.....	27
2.4 Interaction prediction through homologous searching	28
2.4.1 Orthologue data collection.....	28
2.4.2 Interactome data collection	29
2.4.3 Conversion of identifiers to Ensembl Gene IDs	29
2.4.4 Interolog prediction.....	29
2.4.5 Confidence measurements for interologs.....	29
2.4.6 Discussion	32
2.5 Annotation to infer interactions	32
2.6 Prediction of protein annotation using sequence features.....	34
2.7 Finding protein annotations using GenePattern	36
2.7.1 Preprocessing Modules: Arff2Gct, Gct2Arff, ProteinDatasetCreation	37
2.7.2 Prediction Modules: ModECOCTrainTest and ModECOCXValidation	38
2.7.3 Prediction Modules: BlastTrainTest and BlastXValidation	39
2.7.4 Utility Module: CombineOdf.....	39
2.7.5 Discussion	39
2.8 Evaluation of public domain protein-protein interaction data	40
CHAPTER THREE: SIDEKICK.....	44
3.1 Overall organization and purpose	44
3.2 Sidekick runs on most systems	45
3.3 Sidekick uses up-to-date information	46
3.4 Tasks	49
3.4.1 Getting a gene list from a search.....	49
3.4.2 Producing gene-pair list	50
3.4.3 Supporting information for results.....	52
3.4.4 Displaying results.....	52
3.4.5 Workspace of results.....	53
3.4.6 Extracting a new list from a Sidekick result	54
3.5 Enrichment.....	55

3.5.1 Enrichment for GO terms.....	56
3.5.2 Enrichment for disease terms.....	57
3.5.3 Enrichment for gene expression.....	59
3.5.4 Enrichment for chromosomal proximity.....	59
3.5.5 Enrichment for gene-pairs.....	60
3.6 Combining lists.....	61
3.7 File management.....	62
3.8 Confidence and belief.....	62
CHAPTER FOUR: IMPLEMENTATION OF SIDEKICK.....	64
4.1 Program Details.....	64
4.2 Sidekick use case.....	66
4.3 Sidekick – SideCache communication use case.....	69
4.4 Sidekick extension use case.....	71
CHAPTER FIVE: BELIEF MANAGEMENT.....	74
5.1 Overview of belief and confidence difficulties.....	74
5.2 Raw scores and credibility.....	75
5.2.1 <i>Score-from-source</i> and <i>Score-from-source credibility</i>	75
5.2.2 <i>Source</i> and <i>source credibility</i>	77
5.2.3 <i>Size-of-group</i> and <i>size-of-group credibility</i>	78
5.3 Belief Manager.....	78
5.4 Belief, disbelief and uncertainty.....	79
5.5 Dempster – Shafer Theory and <i>Combined credibility</i>	80
CHAPTER SIX: CASE STUDY.....	85
6.1 Introduction.....	85
6.2 Sidekick compared with InterologFinder.....	85
6.3 Typical manual approach (as illustrated by InterologFinder).....	87
6.4 Sidekick's flexible exploratory approach.....	88
6.5 Example of Sidekick's handling of belief.....	89
6.6 Discussion.....	89
6.6.1 Data downloads and processing.....	89
6.6.2 Inflexible scoring.....	90

6.6.3 Simple use of NCBI.....	90
CHAPTER SEVEN: CONCLUSION AND FUTURE WORK.....	92
7.1 Conclusion	92
7.2 Extending Sidekick.....	92
APPENDIX.....	94
BIBLIOGRAPHY.....	95
Vita	

LIST OF TABLES

Table 1.1 Typical Research Questions.....	3
Table 1.2 Sidekick's key features.....	6
Table 2.1 Confidence measures for YBR072W (a yeast protein)	34
Table 2.2 Some entries from ModECOC and WolfPsort confusion matrix for the plant dataset.	35
Table 2.3 Unique features of human PPI databases.....	41
Table 3.1 Data providers and their benefits	48
Table 4.1 General description of Sidekick's top level classes	65
Table 4.2 Analysis steps for disease →gene list.....	66
Table 4.3 Analysis steps for genes → interacting pair list	67
Table 4.4 Web service call steps and implementation description	69
Table 4.5 Steps to create a new query.....	71
Table 5.1 Basic probability assignments, beliefs and subsets	81
Table 5.2 DST combination of stronger credibilities.....	83
Table 5.3 DST combination of weaker credibilities	83
Table APPENDIX.1 Query classes required by QueryBaseADT interface	94

LIST OF FIGURES

Figure 1.1 Cytoscape screen shot.....	7
Figure 1.2 Computational steps of an analysis	16
Figure 2.1 Signal transduction pathway of EGF from KEGG.....	24
Figure 2.2 MiPPS GenePattern workflow	36
Figure 2.3 Overlap of PPIs and proteins in human PPI databases.....	42
Figure 3.1 Sidekick screenshot.....	45
Figure 3.2 Query screenshot.....	49
Figure 3.3 gene list screenshot.....	52
Figure 3.4 Sample Sidekick workspace.....	53
Figure 3.5 Icon screenshot.....	53
Figure 3.6 Extracting screenshot.....	54
Figure 3.7 Enrichment screenshot.....	55
Figure 3.8 AdvancedDataGrid screenshot.....	55
Figure 3.9 Combine options screenshot.....	61
Figure 3.10 Manage options screenshot.....	62
Figure 3.11 Scores and credibility screenshot	62
Figure 4.1 Top level Sidekick classes.....	65
Figure 5.1 Score-from-source screen shot	75
Figure 5.2 Individual result screen shot.....	78
Figure 5.3 Screenshot of Belief Manager	79
Figure 6.1 Comparison of research goals reached by hand and by using Sidekick.....	85

CHAPTER ONE: THE NATURE OF BIOLOGICAL KNOWLEDGE DISCOVERY

1.1 Introduction

Increasingly, the search for mechanisms and biological processes in complex diseases begins with exploration of data from many sources to incorporate clinical, molecular, and high-throughput genomic data. A scientist might search literature and other databases for information about specific genetic elements and their associations with other elements; using this information to infer information about new elements and associations. Only after forming hypotheses about likely candidates for study does the scientist take the work to the wet lab for validation. The discovery process requires downloading data from several data sources, matching identifiers between data lists, and manipulating lists to match elements from one list with elements from other lists. In addition, the research might involve finding relationships between elements both within and between lists. Further, results that have scores might provide a measure of quality. During several research projects including protein annotation prediction and protein-protein interaction prediction, the challenges of the discovery process became evident to me. Existing computational methodologies that I emulated to aid biologic discovery lacked flexibility. Furthermore, while developing algorithms and tools with various biological scientists including my work with scientists at the Greehey Children's Cancer Research Institute (GCCRI), I encountered a divide between the goals of biological research and tools that enable reaching the goals. This divide was the genesis for Sidekick.

Sidekick is a web-based genomic decision framework that bridges the work of laboratory and computational scientists. Sidekick's knowledge discovery strategy focuses on bottom-up discovery and organization of belief. Sidekick combines multiple sources of data for many common research tasks including determination of genes involved in a disease, diseases associated with a gene, gene expression enrichment, Gene Ontology enrichment, chromosome locality enrichment, and interactions. By using web services, Sidekick keeps its information as current as the data sources themselves. The user can save and combine analysis steps in order to easily document and reproduce results or back track to previous states when investigations in one direction do not produce meaningful results.

Explained in depth in Chapter 6, the scientists at the GCCRI accomplished protein-protein interaction prediction using known interactions from one species to predict interactions in

another species. For a single pair of proteins, anyone with a web browser could visit five different websites and determine the possibility of an interaction between the proteins with a paper, pencil and an hour of time. The supporting information like number of species, data sources, and evolutionary strength of the connections between species could be noted and influence belief in the prediction.

For the genome wide analysis called InterologueFinder [1], the computational work included data downloads, both manual ftp and using Perl scripts, as well as the creation and use of a Java program to find connections between species and generating scores. Modifications to the program were made when there was disagreement between researchers about scoring formulas. This required rerunning the program for all species. Data from data sources was updated during the course of the development, which required fresh downloads and rerunning of the program. From the beginning of the work, the biological scientists knew the steps involved in the research; however, the development of the tools to accomplish the research took much time and effort to accomplish the genome-wide analysis. Many compromises on the final scoring mechanism were made because of biases among individuals, and these compromises were permanently encoded in the algorithm due to the offline nature of the approach.

As is typical for bioinformatics work, InterologueFinder makes its results available through a website in which users can enter genes of interest and retrieve the predictions for their genes. This is possible because InterologueFinder has precomputed the results for all possible genes of interest. However, the results were obsolete long before the paper describing InterologueFinder was published because many of the underlying datasets on which the computations were based are updated weekly or monthly. In order to keep InterologueFinder up-to-date, the researchers would have to regularly devote time for a computational scientist to repeat the calculation.

Sidekick takes a fundamentally different approach to knowledge discovery from typical bioinformatics tools by using timely information during discovery. During the development of Sidekick we addressed all of the requirements of InterologueFinder and more. Sidekick enables queries and combination of complex data sources to facilitate research and discovery with no requirement for computational expertise. Large-scale interaction prediction is accomplished with only a browser. Sidekick capitalizes on web services provided by quality data sources to maintain up-to-date information from a variety of web sources so that when the prediction

analysis is performed, the data is current. Sidekick manages the creation of gene pair lists representing both interaction and orthologous and the combining of these lists for interaction prediction using other species. The application not only incorporates scores for individual calculation, but it also facilitates incorporating the researcher's beliefs to weight the results.

1.2 The bioinformatics research process

Table 1.1 lists some common research questions that require either wet-lab experiments, data base searching, using a tool to manage a particular analysis step, or a combination of these tasks with compilation of results.

Table 1.1 Typical Research Questions

1. What genes are involved in a particular disease?
2. What diseases involve this gene?
3. What are the products of these genes?
4. What are the functions of the orthologs of these genes?
5. What are the known mutations involved with a particular disease?
6. What genes are differentially expressed between a disease and normal condition?
7. What diseases have common expression levels for these genes?
8. What are the targets for a transcription factor?
9. What interactions are the proteins of a group of genes involved in?
10. What genes of a group of genes localize to the same cellular component?
11. Is it possible that this group of genes were mutated at the same time?

1.2.1 Manual database search

Typically, bioinformatics data analysis follows one of two strategies. A top-down approach draws generalization from large genome wide data analysis while a bottom-up approach takes known genomic data, perhaps in the form of a single gene's function, and explores relationships between the known information and the data directly related to it [2]. Obviously, each approach has advantages and disadvantages and utilizing only a single approach limits solving research questions.

Wet lab scientists typically approach the questions of Table 1.1 using bottom-up techniques. For example, to answer questions such as "What genes are involved in a particular disease?" "What diseases involve this gene?" or "What are the products of these genes?" a scientist might prefer a simple web search to find links to documents in NCBI's PubMed [3] that provide publications containing the search terms. By reading the articles, a scientist could discover additional related information. Utilizing reliable sources that have pre-parsed the

information is important, given the sheer amount of literature present in PubMed. NCBI's Entrez gene has compiled gene information to allow discovery of disease information given a gene. NCBI's Online Mendelian Inheritance in Man (OMIM) provides information about a specific disease including genes involved in that disease.

Other web searches could answer other questions. Information from one species can offer insights on the species of interest. Genes from different species are orthologous when they evolved from a common ancestor's gene. Ensembl [4] includes orthologous information between species and the degree of gene similarity after speciation events. If the orthologous species is a well-studied one such as the fruit fly, functions of well-conserved fly orthologs can give insight on the functions of orthologous human genes.

Complex questions about specific genes or proteins might be answered within a research paper. For instance, a paper on the PHOX2B transcription factor might indicate that the factor targets the TLX2 gene. Assembling this type of information from numerous sources and from databases such as TRANSFAC [5] would provide potential answers to question 8 of Table 1.1. However, it is also possible the information does not exist and cannot be found by a literature or database search. Either a tool will be used or computational skills will be needed to infer unknown information from existing information. In the example of transcription factor targets, a predictive algorithm might exist or an orthologous transcription factor / transcription factor target pair in one species might offer insight into orthologs in another species.

1.2.2 Research by using tools

To address the tedious nature of hand searches, much of the previous work in biological knowledge discovery has focused on large-scale top-down discovery rather than bottom-up development of hypotheses. For example, Castellano *et al.* [6] mine information from 5,000 scientific documents using parallel processing in a grid computing environment. Their sample application extracted symptoms and pathologies from the unstructured documents, highlighting the computation power of a grid approach for large-scale discovery using text mining. While their middleware solution tool could be used to explore answers to the first six questions of Table 1.1, the scientist interested in a particular gene or particular disease would need to sift through the results to find relevant relationships.

Pounds *et al.* [7] have developed a tool that determines the statistical significance within groups of gene expression datasets by identifying patterns of association with more than one endpoint analysis. G-SESAME [8] determines gene similarity based on Gene Ontology (GO) [9] terms, while ClueGO [10] and PIPE [11] facilitate mass spectrometry analysis and gene annotation exploration. All of these were developed in a top-down genome wide manner, but are helpful for research involving the specific annotations.

Websites like NCIBI's MiMI [12], KEGG [13], EMBL [14], STRING [15], and DAVID [16-17] provide very useful annotation information with the benefit of an easy to use web interface. However, they still require effort to combine and compile discoveries.

An obvious step is to combine multiple focused research tasks into a single tool for knowledge discovery. The DiscoveryNet system [18] facilitates this combination using grid computing for computationally expensive analyses. The system allows the user to build reusable workflows that can be deployed for use outside the original creator's lab. DiscoveryNet has become an important portion of the InforSense consulting company, which specializes in high-throughput discovery workflows. In one example, cited on InforSense's website, Celera Corporation stated they used an InforSense workflow to browse, analyze, and integrate clinical data including enzyme linked immunoassay and single nucleotide polymorphism data. While DiscoveryNet is a viable solution for large companies, the software appears to be financially out-of-reach for smaller labs and appears to limit the knowledge exploration to pre-determined workflows rather than allowing the incremental discovery of information to drive the discovery process.

Gaggle [19] in conjunction with Firegoose [20], on the other hand, provides a free plug-in for the Mozilla Firefox web browser that facilitates transfer of information between various bioinformatics websites including KEGG, EMBL, STRING, and DAVID. Users can transfer information to and from local desktop applications such as PIPE and ClueGO that perform specific bioinformatics analyses. Although this approach offers the flexibility of no set workflow order and free availability, the tools require multiple installations and provide no visual representation of the steps required for a particular analysis. Users must parse the output from each web site source. Also, the system does not provide a mechanism for assessing and organizing the user's belief or confidence in the results.

GenePattern [21] runs within a web browser but also can be downloaded and run locally. Originally created for gene expression analysis, GenePattern also enables single nucleotide polymorphism and proteomics analysis. GenePattern provides fixed workflows that encapsulate analysis processes and allows the development of user-created workflows built from existing modules. GenePattern processes are oriented towards capturing detailed and possibly large-scale computational analysis rather than initial exploration, knowledge discovery, and evaluation of data.

QuExT [22], which focuses primarily on literature searches, retrieves relevant articles given an input set of genes and modifies the order of article relevance to reflect user belief. It initially gives each synonym and the gene name equal weights; however, the user can indicate preference for article types by increasing the weight of that synonym or the gene name concept, thus resorting the results to match the user's belief.

1.3 A comparison of Sidekick, InterologueFinder, Cytoscape and DAVID

The key features that set Sidekick apart from other bioinformatics research tools are outlined in Table 1.2. Two important and widely used biological tools mentioned in the previous section are DAVID [16-17] and Cytoscape [23-24]. In this section, Sidekick's key features are compared with the features of DAVID and Cytoscape. The challenges of developing InterologueFinder are also described as they relate to the key features.

Table 1.2 Sidekick's key features

<ol style="list-style-type: none">1. Transparent integration of information2. Belief management3. Pair-wise analysis using sets not graphs4. Workspace management5. Automatic updating of information6. Bottom-up development of modules for extensibility

Cytoscape is an open source program that is downloaded and installed on the user's computer. It was primary constructed to visualize biological interaction networks and enhance each node with annotation information including gene expression data. Figure 1.1 shows an example of a Cytoscape network. The user can load a large network with 10,000+ nodes and edges. They can change the layout and appearance of the network and zoom in on parts of the network.

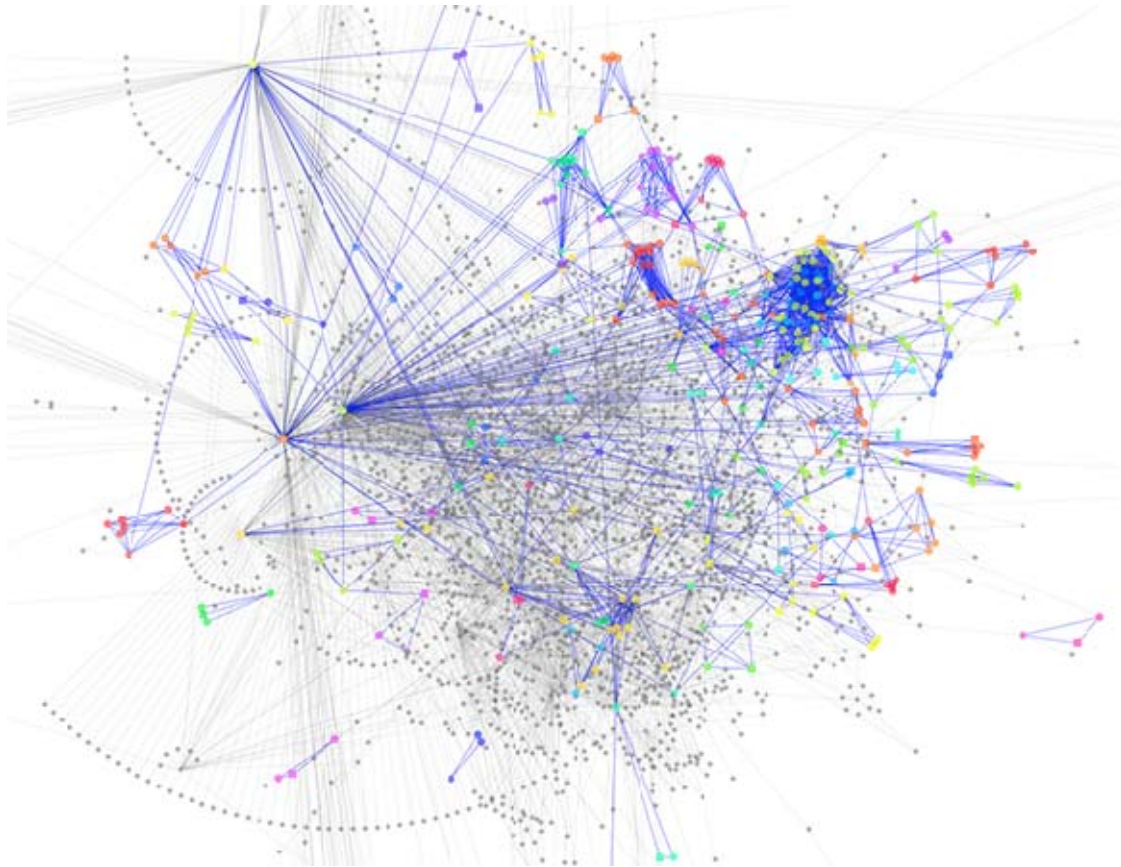


Figure 1.1 Cytoscape screen shot

Cytoscape also allows users to develop program plugins or use plugins developed by other users. The plugins include various modules to perform analysis on existing networks as well as modules to import data from data files and to build queries for web sources, network inference plugins such as Cytrophet [25] allows user to infer new networks. Cytrophet predicts protein-protein interactions using three different algorithms. The user imports their list of proteins and runs the analysis to find a predicted network of interactions. Function enrichment plugins highlight subgraphs that contain similar annotations. Finally, the communication / scripting plugins allow the user to connect to other bioinformatics tools.

DAVID, the Database for Annotation, Visualization and Integrated Discovery, is another well known bioinformatics annotation suite of tools. Unlike Cytoscape, DAVID runs within a web browser. DAVID contains five tools: the DAVID Gene Functional Classification Tool, the DAVID Functional Annotation Tool, the DAVID Gene ID Conversion Tool, the DAVID Gene

Name Viewer, and the DAVID NIAID Pathogen Genome Browser. The massive repository of information behind DAVID provides information on tens of millions of biological identifiers. DAVID now provides a comprehensive set of functional annotation tools for investigators to understand biological meaning behind large list of genes. For any given gene list, DAVID tools are able to enrich and group genes by specific annotations, visualize genes within pathway maps, display genes and matching terms in a heat map, search for genes given a function, list interactants and diseases for genes, list protein domains and convert gene identifiers into various forms.

1.3.1 Transparent integration of information

Often matching identifiers is difficult due to differences in naming conventions and standards for renaming and retiring identifiers. For InterologueFinder combining three data sources required writing a simple Java program to compare each identifier against a translation file. To ensure current information, NCBI was queried for each identifier to determine if the gene was still valid. Cytoscape offers a plugin to the NCBI eUtilities web service, however the user must form the query which requires knowledge of NCBI's query format. DAVID allows the user to enter a variety of identifiers and has a powerful conversion tool for selecting specific naming types. Once the conversion is finished, the user copies and pastes the converted identifiers into the appropriate analysis. Sidekick hides the integration from the user by using NCBI gene IDs and gene names as common identifiers. For any query task, Sidekick represents all results in NCBI form. When Sidekick takes data from multiple sources, conversion to a single type of identifier occurs with no user input. When a user imports their own list, they must use NCBI format, however either gene ids or gene names can be used.

Currently Sidekick does not provide the ID conversion services that DAVID provides. However, DAVID, like many of the best bioinformatics tools, makes its underlying database of conversions available for bulk download. DAVID also has an alpha version of a web service API, offered with the restriction that no more than 200 hits/day from one computer with an interval of 10 seconds between hits. According to the DAVID website, most of the data underlying the DAVID knowledgebase was last downloaded in September of 2009. The Sidekick framework infrastructure would allow the incorporation of David's ID conversion capabilities with very little work.

Within the steps of an analysis, data generally flows from step to step. The form of the data changes with each step. In the case of InterologueFinder, one such step took the set of genes in one species and determined the set of orthologous genes in another species. A subsequent step of InterologueFinder uses the resulting set of orthologous genes as input and finds the set of genes known to interact with each of the genes in the orthologous set. A final step connects the results to find the connections between the two results. All of this was accomplished in a Java program that handled the generation and combination of results.

Cytoscape models the results of each step as a graph. Although Cytoscape does not currently have an orthology plugin, such a plugin to find interactions would produce a resulting graph that contains nodes representing the genes and edges representing the interactions between the genes. Assuming there was a way to construct a graph representing orthologs, Cytoscape provides no easy way to combine different types of graphs. DAVID focuses on the annotation of each gene, so it could find the set of orthologs and the set of interactions for each gene; however DAVID also has no feasible way to integrate the information. Sidekick uses sets rather than graphs as the fundamental underlying representation, which allows mathematical operations such as union and intersection as well as algebraic relationships such as transitivity. Through queries and combine modules Sidekick allows discovery and combination of results in a simple visual manner of selecting icons and clicking the appropriate combine operation.

1.3.2 Belief management

InterologueFinder's complex scoring mechanism for finding a single score that describes the quality of any given interactions is fully outlined in Chapter 2. The scoring mechanism combines several values including the strength of the orthologs between species, the number of species that support the interaction and the strength of interactions in terms of experimental support. It does not allow flexibility for changing the weight of any one measurement, so the number of species counts the same towards the final score regardless of a user's belief in the validity of using number of species as a meaningful predictor of quality. DAVID and Cytoscape display annotations including scores from the data sources, however, they do not combine scores and do not integrate user belief in specific measurements. An excellent example of weighting evidence is provided by String [15]. For each interaction, String includes a variety of measurements that quantify the support for the interaction. However this is only for interaction

information, and except for the ability to include or discard specific scores, the user cannot adjust weight based on their belief in the quality of any given measurement.

Bayesian methods have often been utilized for combining multiple sources of information into a single score. This method has been used for interaction network scoring. Xia *et al.* [26] use a Bayesian Network that combines a variety of experimental databases to create a comprehensive interaction network. They created an online database with the ability to visualize and query the results. The product of their work is called IntNetDB. The data sources included 3 gene expression; protein-protein interaction sources from 5 yeast, 1 worm, 2 fly and 2 human studies; the GO dataset for yeast, worm, fly, mouse and human; 2 yeast and 1 fly phenotypic datasets; 2 yeast genetic interaction datasets; 1 domain-domain interaction dataset; and 3 yeast gene context datasets. These datasets totaled 27 datasets. Each dataset was reduced to a collection of probabilities to define the chances that an interaction occurs between two proteins. First they determined two datasets as the gold standard positive (GSP) and the gold standard negative (GSN). Gold standards were needed in the formation of the probabilities used by their Naïve Bayes classifier. They used the likelihood ratio calculated as $\Pr(E|GSP)/\Pr(E|GSN)$ where the $\Pr(E|GSP)$ is the probability of a certain evidence observed within GSP set and $\Pr(E|GSN)$ is the probability of a certain evidence observed within the GSN set. GSP was set from the HPRD dataset described earlier and GSN was the interactions between all possible pair combinations of proteins assigned with a location of nucleus (2224 proteins) and plasma membrane (1397 proteins). Proteins in a different locations were annotated as not interacting. The authors considered the set of 3,106,928 interactions to be adequate for covering interactions of low prediction probability.

Using the likelihood ratio, they evaluated the predictive quality of each dataset taken individually. Each of the 27 datasets was shown to offer a high enough likelihood ratio to be considered in the comprehensive network although they discovered much variation between the best and worst datasets. They integrated the datasets through simple multiplication of the likelihood ratios of the diverse evidences. The resulting Naïve Bayesian classifier was created efficiently and uses the individual probabilities to construct global probabilities for each pair of proteins forming the network.

One flaw that is evident with the discussion on data sources is the belief that HPRD can be considered a gold standard for positive interactions. Although it contained the most

interactions from the group of datasets studied by Mathivana *et al* [27], its lack of overlap with the other datasets indicates its coverage should not be considered as a gold standard. If gold standards are considered unreliable, then the assignment of probabilities become infeasible and other values must be used. In IntNetDB expression data and GO annotations might be counts, domain-domain scoring might be a measurement of surface area between matching domains and yeast interaction datasets a percent representing similarity between human and yeast proteins. Naïve Bayesian requires probabilities and without some conversion, these values are not appropriate. Further, this methodology assumes each measurement to be as useful as the others.

The IntNetDB website provides a link for a bulk download of the data table that resulted from this complex calculation. Using Sidekick's infrastructure, we could easily write a web service to extract information from this table in an online fashion and integrate it into Sidekick. The score column of the downloaded table could easily be incorporated into Sidekick with very little programming as described later in the thesis. The IntNetDB website does not give an indication of when the underlying data was originally downloaded or when it is scheduled to be updated. The website lists a copyright of 2008.

Sidekick's Belief Manager was developed to address the need for a scoring methodology that enables user's biases about the quality of information to influence the final quality score. Further, it addresses the problem of requiring probabilities for Bayesian methods. Explained in depth in Chapter 5, the Belief Manager uses a visualization module that capitalizes on ranking and clustering to convert scores of any format into a confidence measurement. This measurement manages to encompass user belief while standardizing scores for combination.

We also recognized that lack of information does not equate to negative information. Instead of Bayesian combination, Sidekick utilizes Dempster Shafer theory [28], which elegantly manages lack of information as uncertainty and not as negative evidence. Dempster Schafer theory combines the user assigned confidence measurements into a single score. In the context of InterologueFinder, this means when scientists disagreed about the validity of different measurements, their individual beliefs could easily be incorporated by Sidekick in the Belief Manager.

1.3.3 Pair-wise analysis

Protein-protein interaction analysis is one example of finding connections between entities. In InterologueFinder, two distinct connections were required to perform the interaction prediction. First, the orthologous relationship between a gene in one species and a gene in a different species connects the genes. Within a species, the known interaction between genes is also a connection. Typically, like Cytoscape, these connections are represented by graphs. We recognize when the final result is represented by a graph the visual representation can give the user a clear idea of the connections between non adjacent nodes, however the network also can become unmanageable in its size and complexity. A good example is the screen shot from Cytoscape's website in Figure 1.1. The graph representation is also difficult to manage as an intermediary result. Combining graphs, especially ones with different types of connections like those from InterologueFinder, is difficult to manage conceptually and to implement computationally.

Sidekick represents a connection between entities as a set of pairs, in the case of InterologueFinder simply the set of orthologous genes and the set of interacting genes. While this does not offer the visual representation of a graph, it greatly simplifies the combination of results. For instance in the InterologueFinder example to combine these sets is accomplished in one step using the *Combine translate* operation. Pathway analysis is an extension to protein-protein interaction analysis where a series of interactions are all related to a single cellular function. Modeling these pathways is important and graphical representations are easy to see; however, the process is still manageable within Sidekick through multiple translations between gene pair sets.

Enrichment finds subsets from a group of entities that are related to each other according to some similar annotation. Enrichment for pair entities can give additional information to a user for pair results. In the case of the interactions discovered by InterologueFinder, Sidekick could be used to find further evidence in the validity of the interactions in terms of shared diseases, gene expression, chromosomal proximity, and GO annotations. Sidekick is unique for its ability to apply enrichment analysis to sets of pairs using a built-in operation.

1.3.4 Workspace and analysis management

Most tools that provide a visual representation of a workflow represent the workflow by a directed graph in which each step is a node and the edges represent the data flow through the workflow. Figure 2.2, for example, shows a GenePattern [29] workflow that we developed to predict cellular location by two separate classifiers using the protein's amino acid sequence as the input. (See Chapter 2 for a detailed description.) The resulting predictions are combined and visualized. An advantage to this system is the ability to utilize the workflow with a user-supplied protein dataset and expect results similar to those found on the testing data set. GenePattern facilitates saving workflows and the user simply indicates their dataset at the start of the process.

The disadvantage to the directed graph methodology is the lack of flexibility during the discovery process. It is probably rare that a scientist starting a new research task to solve a problem knows all of the steps needed. More likely, the results from one step lead the research to try something else. Finally, through many steps and possible missteps, the researcher finds a solution. Sidekick's facilitates the discovery process by presenting the workflow as a set rather than a graph. Sidekick represents a result from the query, enrichment and combination modules as an icon that indicates which operation was performed to generate the results and what inputs were used. Within the workspace, the user can click icons to return to previous results for input into other operations. The workspace visually depicts what has been discovered.

In the case of Sidekick performing interaction prediction like InterologueFinder, the steps for interaction prediction are sequential, and the resulting workspace does reflect the order of operations. However, the user does not need to perform operations sequentially as long as the data is available, because the icons themselves show where the input came from and hence incorporate order information. A disadvantage of this approach is that Sidekick cannot reproduce workflows that solve specific tasks, such as the one implemented for GenePattern. Currently, the user is able to save workspaces that visually represent steps (as well as the associated data). We plan to implement the ability to save workspaces as set workflows to enable the same analysis steps to process different input data in a future update of Sidekick.

1.3.5 Automatic updating of information

Most data sources have stated policies on frequency of updates to the data. For example, NCBI states that the gene information is updated daily. If a program like InterologueFinder

utilizes data from NCBI, it must perform data downloads daily to maintain up-to-date data. For InterologueFinder, this is not feasible due to lack of automation. Someone would need to manually download data, run the file combination program, and run the predictor program each day! The result is out-of-date information caused by infrequent updates. Cytoscape and DAVID utilize external data sources, and users depend on the maintainers of these sites to update their sites to provide up-to-date information.

Another challenge facing users of NCBI in particular and other web services in general are policies on frequency and timing of queries. NCBI states that if a user exceeds a set number of queries per second or number of queries at a time, their IP address will be identified and future queries will go unfulfilled until the user requests approval from NCBI. Cytoscape enables a user to enter queries to NCBI in one of their input plugins. It is possible that Cytoscape has an agreement with NCBI to accept more than the stated limit. However, it is also possible that if the plugin was used in an automated fashion within Cytoscape, the user would exceed allowable limits and be banned.

SideCache was developed alongside Sidekick to manage information flow between Sidekick and external data sources. Explained in depth in Chapter 3, SideCache manages data in two ways. First, all web service queries go through SideCache. Each web service is configured to perform to specific number and frequency policies (via an XML file). SideCache also maintains a cache of recent results. If an already fulfilled query arrives, SideCache returns the cached results without another call to the external web service. For any web service where it is not feasible to remain within set policies, like NCBI, SideCache enables creation of a web service that runs within SideCache and batch downloads files from the source web service according to an automatic update schedule. The updated data is then automatically exposed to Sidekick's queries. SideCache base classes and configuration files have been developed with careful attention to synchronization issues to allow quick deployment as new data sources become available.

SideCache also adds functionality to Sidekick. Gene enrichment of hierarchical data is accomplished using a strategy outlined in Chapter 3. An open source implementation for this enrichment is available however it requires scheduled downloads of data and due to differences in programming languages could not be incorporated into Sidekick. SideCache provided a

natural means to maintain the data and wrap the source code for the enrichment into a servlet managed by SideCache.

1.3.6 Extensibility

Cytoscape and GenePattern both encourage users to create plugins for their programs. Further Cytoscape, DAVID, and GenePattern welcome requests for new functionality to enable additional plugin development. Sidekick was designed in a bottom-up approach to encourage future expansion. Each query, enrichment, and combination operation follows standards for data collection, display, and file management. This standardized functionality is facilitated by a base class for common methods and data elements and a base interface to explicitly indicate functionality each operation must fulfill. The underlying XML file representation for saved results and workspaces also enables continuity between current and future operation modules.

1.4 Computational overview

Sidekick was built using many computational tasks inspired by several computational topics including machine learning, visualization, distributed systems, data structures, and computer architecture. Sidekick provides rich functionality to the user via an easy-to-use graphical interface. Machine learning algorithms used in Sidekick and the work leading up to Sidekick include Error Correcting Output Code, Decision trees, combining classifiers, Expectation Maximization and K- nearest neighbor clustering. Visualization was used extensively for belief management. For asynchronous communication between Sidekick, SideCache and various web services, distributed system techniques were employed. Caching and optimizing lookups required database and data structures knowledge. Careful consideration of programming languages and architecture aspects of the project resulted in the development of a framework that runs on most systems and allows easy extensibility.

The specific Sidekick operations will be fully developed in Chapters 3, 4, and 5 but to get a sense for Sidekick's organization and inner working we explore the biological question "Do there exist other diseases that are related to Wilm's tumor". Perhaps the user believes that genes that are known to interact with genes known to be involved in Wilm's tumor might give clues to other diseases. Figure 1.2 introduces the computational aspects of Sidekick and the information flow between Sidekick, SideCache, and external data sources using this simple example. The

computation steps are indicated by letters A-D, while the information flow steps are indicated by numbers 1–11.

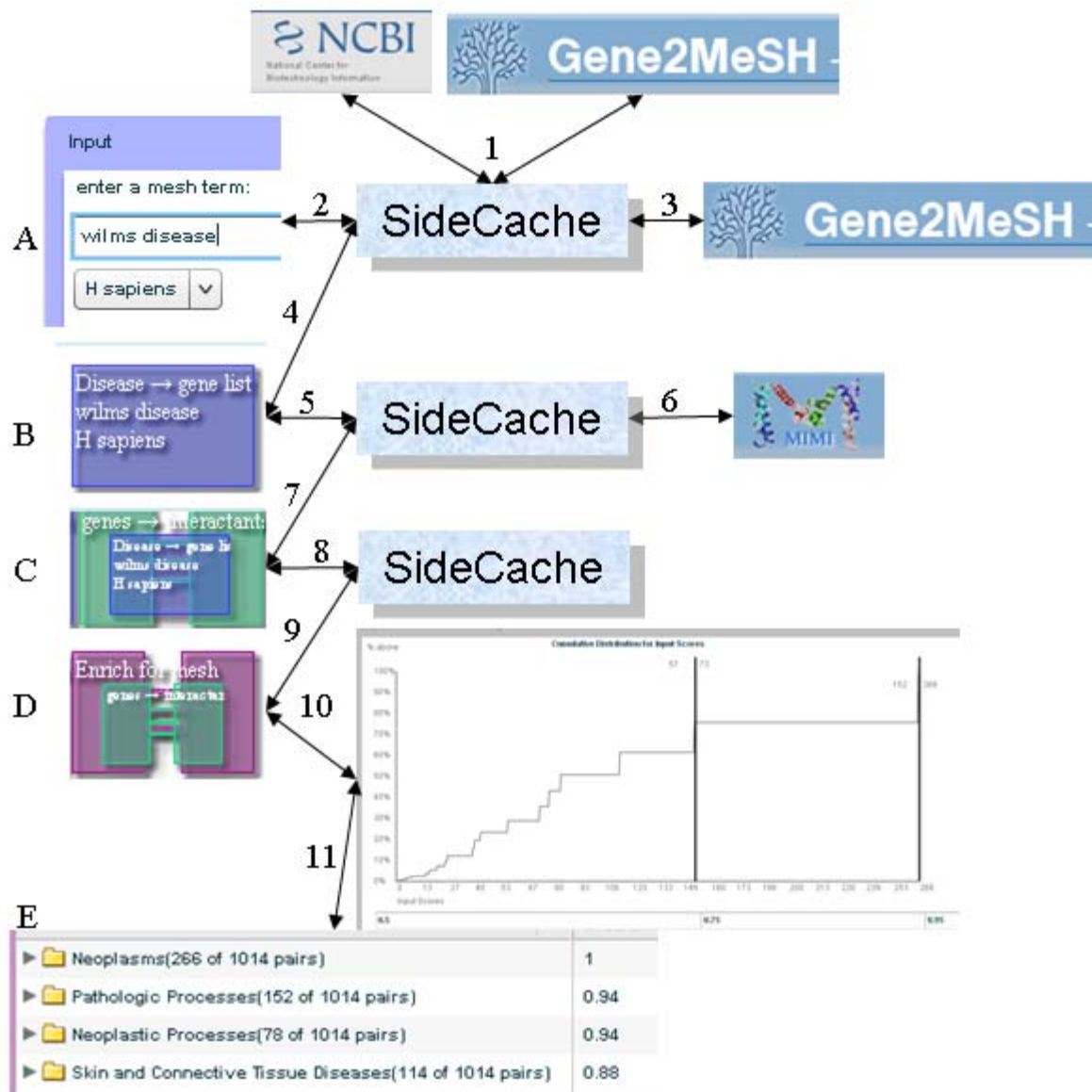


Figure 1.2 Computational steps of an analysis
Numbers indicate information flow and letters Sidekick results

SideCache and the servlets that it manages are initialized when the web server starts up. Once initialized, SideCache waits for queries from Sidekick as well as periodically downloading and processing files to fulfill requests locally.

1. Initial download and processing of gene and interaction information using bulk download has occurred when the web server starts up (and is automatically updated

on a scheduled basis). In addition, previous requests from all Sidekick users that require a web service request from other sites have primed the SideCache cache with previously fetched web pages. Because it is possible that other threads have access to the information careful synchronization ensures current and complete data for Sidekick queries. The data is stored in hash tables to ensure $O(1)$ access times. The creation of the hash tables takes much more time, but the cost is only incurred once daily (scheduled in the middle of the night). NCBI eUtilities and NCIBI's Gene2Mesh are two commonly used services. The NCBI eUtilities information includes interaction, chromosomal, and gene identifiers. The Enrich for Mesh Terms servlet requests and receives disease annotations for all genes from the Gene2Mesh web service of NCIBI.

To start the analysis, the user opens the Sidekick application in any web browser with Adobe Flash 10 installed (<http://visual.cs.utsa.edu/sidekick>). This approach provides machine, operating system, and browser independence. Unlike Cytoscape, Sidekick runs on the local machine without the need for a download and install. The user selects the *query disease to gene list* to start the process.

- A.** The input screen for this query appears within the Sidekick application. Each query is written with similar formats for easy programming and a consistent feel of the program. The user enters *Wilm's tumor* in the input box and runs the query.
- 2.** Sidekick initiates two asynchronous communication threads to SideCache. Each has an active listener to process the returned communication. The first thread requests NCBI gene information and is fulfilled with the previously data downloaded in Step 1.
- 3.** The second requests information from NCIBI's Gene2Mesh via SideCache. SideCache either fulfills the request from cached data or by communicating with Gene2Mesh following access guidelines set up in SideCache's settings file.
- 4.** As the results of the two requests come in, Sidekick fills in its Results table.
- B.** Sidekick creates a result icon in the workspace and stores twenty genes in a flexible graphical data structure. Some of the genes are supplied from NCBI via SideCache and others from Gene2Mesh; redundant genes are combined and noted as from both sources. Gene2Mesh only returns gene names so another communication thread to the

SideCache's NCBI information servlet (not shown in the diagram) is started to fill in the other information. This set of genes forms the input for the *genes to interactants query*.

5. When the user executes the interactants query, Sidekick starts two communication threads using the input genes in the query. The first query to SideCache's NCBI information servlet returns the interaction information, including the additional genetic information from the data managed by SideCache. The second query is to NCIBI's MiMI
 6. As in Step 3, SideCache manages the communication with MiMI and returns the results to Sidekick.
 7. As the results come in, Sidekick fills in its Results table. The information from MiMI only includes gene names, so Sidekick initiates communication with the SideCache's NCBI information servlet and adds the results as they come to the data structure (not shown as a separate step).
- C. Sidekick displays the result icon in the workspace and the resulting non redundant gene pair set with 1014 pairs in the results grid. The user clicks *enrich for mesh*.
8. The enrichment module takes all gene pairs as input and creates a communication thread with SideCache. SideCache's enrichment servlet uses information downloaded in Step 1 and the open source implementation of enrichment using hierarchical data to find the diseases common to some of the input gene pairs. In this example, 20 disease groups are determined each with varying number of gene pairs present, ranging from 2 to 266 gene pairs. The final step of the analysis organizes the 20 groups by their confidence scores.
 9. As the results of the two requests come in, Sidekick fills in its Results table.
- D. Assume for this example that the user believes higher numbers of gene pairs present in a disease group indicates higher confidence that the disease is related to Wilm's tumor. Also assume that lower p-values found for a disease group indicate more belief in the importance of that enriched group. When the user clicks *Adjust size of group credibility*, Sidekick launches its graphical Belief Manager.

- 10.** The Belief Manager graphically displays the values, shown are the number of gene pairs present in a disease group, in a cumulative distribution graph with clusters of values found by the Expectation Maximization algorithm. To increase the number of clusters, the KNN algorithm is used. The user enters confidence values for each cluster, which enables assignment of higher confidence for larger group sizes and lower confidence for smaller group sizes. The Belief Manager uses a similar process for the normalized p-values.
 - 11.** Changes to the final confidence score are reflected in Sidekick's Results grid.
- E.** In the final step of the analysis, Sidekick sorts the result grid associated with the Enrichment of **D.** by combined scores. These scores now incorporate the user's beliefs and two types of scores, counts and p-values, that form the basis for this confidence. Notice that neoplasms have the highest confidence scores. This is to be expected because Wilm's tumors are neoplasms specific to the kidney. However several other diseases terms also have high confidence measurements indicating possible diseases related to Wilm's tumor, our initial biological question. The Sidekick workspace displays each of the results steps providing a representation of the process. Each result can be returned to by clicking the icons in the workspace. If another enrichment is required, the result icon **C.** is clicked and forms the input for that analysis step.

Each module was constructed to meet the requirements of standardization of data storage, web service processing, asynchronous communication, non redundant results, and file management. The specifics are covered later, but in general, each query, enrichment and combination contains either a single gene or gene pair list. The lists are data structures that store instances of genes or gene pairs along with the information specific to the operation. Sidekick stores each gene instance locally in a global gene information data structure (in the client browser) to prevent redundant web service queries. When a query requires genetic information, Sidekick first determines whether the gene identifier exists in the global structure. If so, Sidekick uses the existing instance. Otherwise, Sidekick initiates the web service query to provide extra genetic information. When two web services return the same gene or gene pair, both sources are included in a single non redundant data entry. When combining results for example using union, Sidekick combines redundant entries. File management is accomplished using a standard XML

representation of the data. This allows simple exporting to other programs and importing user created results. The results workspace represents each gene and gene pair list with specific icons. Each includes information specific to the query, enrichment, or combination. The workspace details steps of an analysis and implicitly describes the progression of research tasks used to discover final results.

We incorporate many features found in other tools into Sidekick, but have developed our own unique visualization belief management system and pair-wise processing modules. Unlike the Cytoscape, DAVID, and GenePattern, Sidekick combines multiple discovery steps and displays them in its workspace hiding the work of data integration from the user. Analyses are accomplished in a user driven, flexible manner that encourages new discoveries rather than repeating old ones as seen in most workflows. SideCache was developed to minimize communication costs while automating the process of maintaining up-to-date data. Sidekick was developed in a manner that will promote future expansion of modules and features.

1.5 Overview of this thesis

Ultimately, to form a clear understanding of the genomic information the scientist will combine research results from both tools and from manual database searches. An automated process to accomplish these tasks would be less tedious and error prone, but such automation is difficult unless one uses set workflows. A flexible application that allows a scientist to explore research in directions directed by the results and not using a set pattern is needed. Further, the application needs to integrate results from both manual research and output from other tools.

Collaboration with scientists has given me insight into key biological discovery tasks and into the fundamental challenges of managing and evaluating the huge amount of information available to researchers. Chapter 2 describes these challenges for protein-protein interaction discovery and annotation, a central genomics research task that motivates this work. Chapter 2 also describes the results of my earlier work to develop usable tools that would help non-computationally oriented scientists tackle these problems. Chapter 3 introduces Sidekick and provides an overview of its capabilities and basic design. Chapter 4 provides an in-depth discussion of the Sidekick architecture and addresses a number of implementation and design issues. Chapter 5 describes Sidekick's approach to belief management and the underlying Dempster Shafer theory on which it is based. Chapter 6 presents a case study comparing

Sidekick and InterologFinder's approaches to protein interaction network development. Finally, Chapter 7 outlines conclusions and future work.

CHAPTER TWO: GENES, PROTEIN INTERACTIONS AND DISEASE

The study of the underlying mechanisms of disease fuels discoveries enhancing the health and well being of all organisms. Integral to the mechanisms are genes and how the proteins they produce interact. If a single gene mutates and changes its function, it can have a dramatic effect on the entire organism because the gene affects each of the functional pathways in which it is contained. Each pathway consists of a series of interactions and to fully understand the complex working of a pathway, the interactions must be fully understood.

Much of my early research centers on proteins and protein interactions. This chapter gives a brief overview of the biology of genes, proteins and interactions. It covers various approaches and sources of data available to study them. The chapter also describes some of my preliminary work addressing the issue of how to make the tools and data relevant to research useable for non-computational scientists. Although Sidekick was developed to enable diverse biological research, much of my early work influenced Sidekick's functionality and organization.

2.1 Biology of protein-protein interactions

The central dogma of genetics is the process of creating proteins from the genetic information coded on the chromosomes. The chromosomes are made up of stretches of base pairs collectively referred to as genes. Genes are transcribed into RNAs and then the RNAs are translated into proteins. As stated earlier there are many more proteins than genes. Genes for species classified as eukaryotic actually translate into several proteins through alternative splicing. The proteins themselves go through several levels of organization that further differentiate the proteins. At the most primary structure level, proteins are made up of a sequence of amino acids connected via covalent peptide bonds. Each amino acid is coded by three nucleotides in the RNA collectively called codons. The next level of structure is the configuration of the series of amino acids into either α helices or β sheets, or a combination of the two types of simple structures. The amino acid sequence dictates the formation of these simple structures. These secondary structures fold into tertiary structures and finally, if the single amino acid sequence binds to another amino acid sequence, the resulting complex is referred to as the quaternary structure.

During any of these stages of structural development, proteins can interact with each other. These interactions have time scales ranging from very short to very long. Mintseris *et al.*

[31] defines the kinds of interaction times that take place between proteins. Hemoglobin in red blood cells is an example of a very stable interaction. Formed by α globin and β globin the hemoglobin protein exists as a molecular structure with a very long life. Most proteins do not permanently bind with other proteins in this manner, but rather more loosely associate or "interact" with them. All interactions are mediated by the areas of the protein that actually come in contact with interacting partners. The study of these contacting surfaces is generally referred to as the docking problem. Docking occurs in complexes and between proteins and is characterized by the length of the contact. Transient interactions occur between proteins, but for much shorter times than those of complexes. Currently, the definition of the protein-protein interaction prediction problem has many meanings. Most researchers currently use the prediction of protein-protein docking synonymously with protein-protein interaction. But proteins that interact are actually a subset of proteins that dock. Docking is merely the process of two proteins coming together. But interaction has to do with conformational or configurational changes or both [32]. For instance, a pair of proteins might become a functional unit when they dock together and then interact with a different section of the sequence or activation site. If they dock, but the activation does not occur, then they do not interact.

The problem of protein-protein docking prediction, which may be addressed computationally, could lead to more accurate protein-protein interaction prediction. The obligate interactions between proteins in a stable complex as well as transient interactions between proteins are regulated by protein docking domains, which are areas of amino acids within the tertiary structure of the protein that form a surface in which docking occurs. Due to folding, the amino acids that make up the domains do not have to be adjacent to the other amino acids in the primary structure. These structural interfaces sometimes are associated with different roles in the cell and therefore can contribute information about the proteins involved in cellular roles. The nature of the various interactions makes protein-protein interaction prediction difficult because of the wide spectrum of possible interactions. Very obligate protein pairs cannot exist without each other and may fold and dock simultaneously [33].

Transient interactions could involve interactions between proteins that last a very short time. These interaction types highlight another one of the difficulties in protein-protein interaction prediction and protein-protein docking prediction. Some data sources that focus on protein complexes will not have any information about transient interactions and cannot be relied

upon to provide full coverage of the information to be used for prediction of transient interactions.

Many interaction pathways have been well studied and their interacting partners have been discovered. The interactions taken collectively can be labeled as a single functional entity, where all of the components need to be present and with suitable docking domains for the function to occur. These interacting partners give information that could be useful for docking or interaction prediction among other protein pairs. The KEGG [13] database lists general categories of cellular function including metabolism, genetic information processing, environmental information processing, cellular processes, human diseases and drug development. Each of these categories is further subdivided into subsections and examples within that subsection. For example, under metabolism, is the section energy metabolism with the examples oxidative phosphorylation, photosynthesis, photosynthesis - antenna proteins, carbon fixation, reductive carboxylate cycle (CO₂ fixation), methane metabolism, nitrogen metabolism, and sulfur metabolism. Figure 2.1 shows an example of the signal transduction pathway of EGF from KEGG.

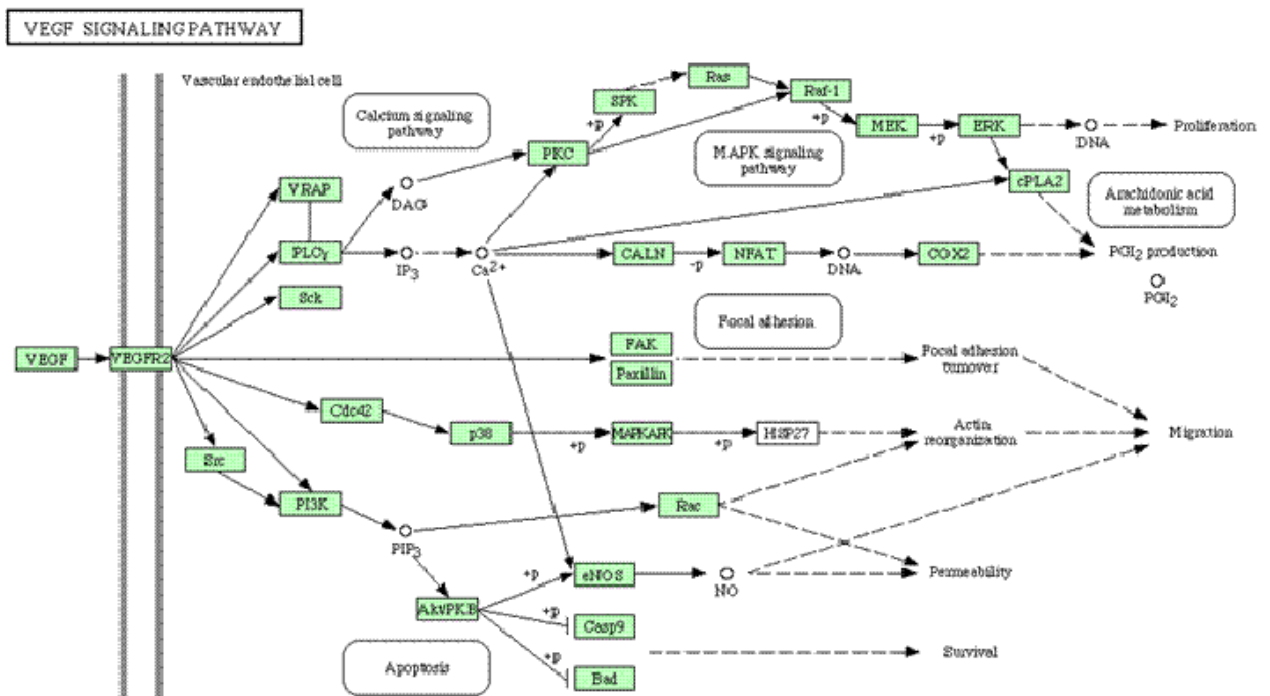


Figure 2.1 Signal transduction pathway of EGF from KEGG

Each square represents an interacting element and each line an interaction. This simple pathway partially shows the activation of the transcription factor that regulates gene expression involved in cell growth. The small molecule VEGF from outside of the cell wall signals VEGFR2 and the signal is carried all the way into the nucleus to influence gene expression, which in turn influences creation of a specific protein. Understanding this functional pathway enables other work, like the discovery of new cancer therapies. For instance, if a tumor suppressing gene was kept from being transcribed, it might lead to cancer growth. Perhaps by blocking the mechanism involved in cancer that prevents the expression of the tumor suppressing gene, the gene would become expressed and the cancer could be reduced.

Researchers have discovered and understood these interactions in new ways. The following sections explain several of the experimental methods that generate data used for various protein-protein interaction and docking prediction methods.

2.2 Experimental methods for determining interactions

2.2.1 Yeast two-hybrid

Yeast two-hybrid is an *in vivo* method of measuring the physical contact between pairs of proteins and can be used for identifying protein-protein docking [34]. In the method, one protein is attached to a binding domain and the other is attached to an activation domain. DNA-binding and activation domains placed together allow for the induction of gene expression. The proteins are placed in a yeast cell and if the reporter gene in the yeast cell is activated, then the two proteins of interest interact because they brought the binding and activation domains together. This method has been used to annotate whole genomes including yeast, worm, fly, and human. It can be used to find both transient and obligate binding. Upon comparing the results from different annotations of the same species (yeast), there was found very little overlap of the interactions [35]. This lack of overlap indicates that the results of this method should not be taken as highly accurate. The inaccuracy could be due to the method's bias towards nonspecific docking, differences in the protein sampling in the experiments, or problems with the method itself. For example, if the binding of the domain to the protein occurs near the docking domain for the protein, this might block the interaction and docking. Also, two proteins might dock when placed together in the yeast cell, but localize to different cellular compartments in their natural environment and not come in contact with each other.

2.2.2 Mass spectroscopy

Mass spectroscopy measures the mass-to-charge ratios of ions. Through this *in vitro* method, individual components in a sample can be determined. These components can then be characterized as protein complexes [36]. The most challenging aspect of mass spectroscopy analysis is targeting the proteins of interest. The proteins need to be isolated and ionized. This targeting makes this technique less useful for discovery of transient interactions, but effective at finding obligate docking and interactions.

2.2.3 Nuclear magnetic resonance spectroscopy

Another high throughput experimental method to determine protein interaction is nuclear magnetic resonance spectroscopy (NMR) [37]. NMR is able to analyze the protein in its natural solution which allows for a closer approximation to the protein as it exists in the cell. However the method is limited to small proteins.

2.2.4 Co-immunoprecipitation

Co-immunoprecipitation is considered to be the gold standard method for protein–protein interaction determination. The known protein of interest is targeted with a specific antibody. When the protein is identified and extracted using the antibody, interactants that are members of the same complex as the known protein can be detected. This method can only identify interactions in which one of the interaction partners is known. Immunoprecipitation experiments can detect transient interactions.

2.2.5 Gene expression

Gene expression refers to the number of proteins produced from a gene. If the environment in the cell prevents a transcription or translation from occurring the corresponding gene is said to be under-expressed. If many copies of a protein are transcribed from a particular gene, then the gene is said to be over-expressed. It is difficult to measure the number of proteins in a sample, but methods have been developed to measure the number of mRNAs which are used to form the proteins. In a study that relates gene expression in terms of mRNA to protein interactions, researchers saw a strong correlation between genes that are co-expressed and protein interaction in complexes [38] and to a lesser degree transient interactions. Gene co-expression is an *in vitro* method that is able to measure transient and obligate interactions with

varying effectiveness, but does not actually measure the physical docking of proteins with other proteins, rather the group of proteins taken as a whole. This important distinction will be significant as prediction of interaction and docking is made. Gene expression experiments are indirect in the sense that they measure the amount of mRNAs present given changes in the cellular environment or during the different phases of the cell cycle. Most researchers use the measure of mRNA as equivalent to the measure of proteins translated by that mRNA, but due to post-transcriptional and translational regulation, it is possible to have the mRNA present, but have no proteins translated from it. This lack of direct correspondence is of concern when predictors are constructed around gene expression data.

2.2.6 X-Ray crystallography

X-Ray crystallography is widely used to determine protein structure [39]. It is an *in vitro* method accomplished by first crystallizing a structure of interest and then bombarding it with x-rays to find the tertiary or quaternary structure. This process gives a clear picture of the shape of a protein complex and the docking domains, but does not show transient docking or interactions. One concern biologists have is the effects of crystallizing the normally fluid protein and how that affects the true structure of the protein and protein complex.

2.3 Computational methods for determining interactions

To increase the coverage of the protein interactome, scientists have developed a variety of computational methods for finding protein interactions. Structural alignment between pairs of proteins can be used for prediction [40-41]. Several machine learning algorithms have been used for protein interaction prediction including Bayesian analysis [42], Random Forest [43-44], logistic regression [43-44], support vector machine [43] and decision trees [43]. When a pair of proteins in one species has been predicted from any of the previous methods and a pair of homologous proteins can be found in a different species, the protein interaction between these proteins can be inferred [45]. Each method has strengths and weaknesses like the experimental methods for interaction determination. The burden will be on the scientist using the data to assess their confidence in the manner in which it was determined. Also if the algorithm provides a confidence measurement in the prediction, the threshold at which the prediction become believable.

2.4 Interaction prediction through homologous searching

In work with Dr. Alex Bishop and colleagues [1], I developed InterologFinder, a tool that predicts protein interactions within one species through the use of homologous pairs in another species. Homology describes the relationship between entities that are similar probably because of a common entity. Orthologous genes are genes from different species that are related by a speciation event. Paralogous genes are those within a species that are related through evolution of the species. Through intra-species and inter-species analysis, we predicted new protein-protein interactions (PPI) for human, mouse, fly, worm, and yeast. We then assigned a confidence measurement to each interaction based on the level of conservation of the interaction across multiple species and the number of supporting experiments. This measurement allowed interologs comprised of one-to-many and many-to-many orthologs to be scored. The resulting protein interaction networks for all five species are available through a website, <http://www.interologuefinder.org>, that allows proteins to be queried. The remainder of this section describes the implementation of InterologueFinder in more detail. This tool is later compared to Sidekick's discovery process during the case study of Chapter 6.

2.4.1 Orthologue data collection

Orthologue data were retrieved from Ensembl 50 for all genes in *Homo sapiens*, *Mus musculus*, *Drosophila melanogaster*, *Caenorhabditis elegans*, and *Saccharomyces cerevisiae*. Ensembl has compared all genes for all species for which complete genome data is available, first against all genes of its own genome and second against all genes of every other genome in their database [4]. We relied on Ensembl for their method of assessing orthology because they used the longest available translation from each gene to run both Blastp and Smith-Waterman sequence comparisons. Relationships between genes were graphed based on best reciprocal hits and best score ratios, before constructing multiple sequence alignment on each cluster that was derived. Phylogenetic trees were then constructed for each gene cluster, and these were fit with speciation trees. Finally, typical pair-wise gene relationships were determined – orthologs of one-to-one, one-to-many, and many-to-many. Additionally, Ensembl provides paralogous relationships both within and between species. These data, along with the percent sequence identities and similarities, were retrieved from BioMart at Ensembl for each of the five organisms investigated.

2.4.2 Interactome data collection

PPI data were retrieved from three databases: IntAct [46], the Database of Interacting Proteins (DIP) [47], and the Biomolecular Interaction Network Database (BIND) [48]. Data was retrieved on Aug 28th, 2008 from all databases, and experimental evidence annotation was obtained when available.

2.4.3 Conversion of identifiers to Ensembl Gene IDs

Synonym tables provided by Ensembl were used to convert the identifier for each protein used from IntAct and DIP into an Ensembl Gene ID. BIND, however, does not provide an identifier present in the Ensembl tables, but provides NCBI Gene Identifier (gi) numbers. We therefore queried NCBI [49] to retrieve the newest gi, and subsequently the associated NCBI protein accession numbers, for each protein. Because we were only interested in PPI, we omitted interactions involving non-protein molecules provided by BIND. For all databases, we selected only those interactions between proteins of the same species using the taxonomy identification provided. For each species, a single file with non-redundant experiments supporting the interactions was created.

2.4.4 Interolog prediction

For those PPI where orthologs exist for both interacting proteins in another species, but no interaction is known between those proteins in that other species, a conserved interaction (an interolog) was predicted. This was accomplished along with statistical data collection using a complex Java program.

2.4.5 Confidence measurements for interologs

To achieve a single, unified confidence score for interactions and predicted interactions, we developed several scores taking into account different attributes of an interaction and combined them to form a single InteroScore.

A protein-protein interaction network is an undirected graph representing all protein-protein interactions (PPI) that occur in a species. Let $F(U,C)$ and $G(V,D)$ be PPI networks representing two different species. Here orthologous relationships between species can be represent by a complete, directed weighted bipartite graph $H(U,V,E,w)$. Here U is the set of proteins from one species and V is the set of proteins from another species. E is the set of all

edges between nodes in U and nodes in V . Each edge is assigned a weight represented by the function w which encapsulates the strength of the orthologous relationship between its two protein nodes as measured by the percent sequence identity. Note: $w(u_i, v_i) \neq w(v_i, u_i)$ because the length of u_i and v_i may be different.

OrthoScore: In order to obtain a score that represents how similar a protein is to its orthologs; all orthologs of a protein were combined to form a single score. This score is adjusted to represent one-to-one and one-to-many relationships between the proteins.

$$OrthoScore(u) = K \sum_{v_i \in V} (w(u, v_i))^2$$

where K is

$$K^{-1} = [\max(w(u, v_i))]^2; v_i \in V$$

For the one-to-one proteins, the $OrthoScore(u) = 1$. This score, however, does not encompass evidence from multiple orthologs in other species, which exist in a one-to-many relationship. Because the identity score for each orthologous relationship can vary, they should provide evidence for a predicted PPI but proportional to their percent identities as compared to the relationship with the highest identity. As an example, the percent identities for three orthologs may be 92%, 87%, and 72%. The highest identity is adjusted to 100%, while the other two are scaled relative to 92% (94.5% and 78%, respectively). The values are squared to give low scores less of an effect than high scores. The resulting *OrthoScore* is the sum of these percents squared ($1.0^2 + 0.945^2 + 0.78^2 = 2.501$).

SingleSpeciesScore measures the strength of a predicted interaction from a single species.

$$SingleSpeciesScore(u_i, u_j) = OrthoScore(u_i) \cdot OrthoScore(u_j)$$

For example, an interaction predicted in fly is known in human with both fly proteins having a one-to-one relationship with their orthologs in human. Each *OrthoScore* will contribute 1.00 to the score because each is both the maximum and the only value. The combined score is 1.0 ($1.00 \cdot 1.00$) representing the support of the interolog. However, given the more complex one-to-

many example where the *OrthoScore* = 2.501, combined with a one-to-one *OrthoScore* = 1.00, the resulting *SingleSpeciesScore* = 2.501 = 2.501·1.00.

During the computation of this score for simplicity we use a score that combines all of the orthology information for both u_i and u_j , however all of the orthologs of these proteins might not interact with each other.

SpeciesScore: The *SpeciesScore* representing the conservation of the interaction is a combination of all of the Species' *SingleSpeciesScores*. Interaction scores were not penalized for their apparent absence in another species in which the orthologs were conserved but no interaction known. In this five species analysis, a predicted interaction from only one-to-one orthologous, with the highest degree of confidence would therefore have a score of 4 (four other species have orthologs with an experimentally verified PPI). Of course, if a single *OrthoScore* is greater than one, the overall *SpeciesScore* can exceed 4. For example, an interaction predicted in fly is known in human, worm, and yeast, with both fly proteins having a one-to-one relationship with their orthologs in each species. Each species will contribute 1.00 (1.00^2) to the score because each is both the maximum and the only value. The final *SpeciesScore* would be 3.00 ($1.00 + 1.00 + 1.00$) and reflects the number of species supporting the interaction. A more complex example is an interaction predicted in fly that is known in worm as a single interaction (again, with all proteins being one-to-one orthologs), but is present in human as three interologs due to paralogues in human (in a one-to-many orthologous relationship with fly proteins). The fly interaction would therefore score 1.00 from worm, but higher from human. Using the one-to-many example above, the final *SpeciesScore* for this second fly predicted interaction example is 3.501 ($1.00 + 2.501$). Contrasting these two examples of high support for the predictions, one is due to the diverse number of species that support the prediction and the other due to the number of interologs within a single species that support the prediction.

ExperimentScore: Using the experimental evidence available from DIP and IntAct (experiment identifier and type), the *ExperimentScore* was developed. For each predicted protein-protein interaction across all species, each type of experimental evidence was only counted once per species regardless of the number of databases in which it was found, and the total number of experiments for each PPI is the *ExperimentScore*. BIND does not include experiment type in its publicly available database and was therefore generally excluded from this

confidence score; PPI annotated only in BIND were described as “protein_protein” and given an *ExperimentScore* of 1.

ExperimentQualityScore: To quantify the quality of various experimental evidences, we determined how many interactions annotated or predicted by a specific experiment were also present in STRING [15]. For a given experiment, such as MI:0019(coimmunoprecipitation) in worm, we determined that out of 4324 interactions annotated by this experiment only 813 of these interactions were present in the STRING database for a ratio of 0.188 (813/4324). Conversely, 4287 interactions out of 5971 MI:0398(two_hybrid_pooling) annotated interactions were found in STRING for a ratio of 0.718. To incorporate as much evidence as possible while adjusting for quality of experimentation support, the ratios for a single interaction's experimental evidence were summed. Thus, an interaction described by both MI:0019 and MI:0398 would have an *ExperimentQualityScore* of 0.906 (0.188+0.718).

InteroScore: To create a single, combined confidence measurement, we took the product of the SpeciesScore and the ExperimentScore then added the ExperimentQualityScore for each interaction and refer to it as the InteroScore.

2.4.6 Discussion

InterologFinder provides a tool that enhances the known interactome with predicted orthologous interactions. InterologFinder includes a confidence score based on experimental evidence, species that support the interaction, and the strength of the orthologous relationship between protein pairs. The information is available for browsing using the InterologFinder website. There is no web service available from the website, and we only cover protein information. The confidence scores can help the scientist who accepts this method for scoring; however the scores are pre-computed before inclusion the website; if the scientist does not have confidence in this methodology they need to disregard the scores.

2.5 Annotation to infer interactions

Annotations of proteins have also been used for protein-protein interaction and docking prediction. For example, shared biological process annotation between proteins can indicate that they interact [50]. This is an *in silico* method of finding data for prediction. The Gene Ontology Consortium (GO)[9] has labeled many of the known proteins with location, function and biological process information. If a GO term is shared by two proteins, then they are more likely

to interact than randomly chosen proteins. This obviously does not guarantee that the protein physically dock with one another, and certainly does not indicate whether they form a complex, but it does provide a means of grouping like proteins and has been used to test predicted protein – protein interactions.

GO terms are organized in a hierarchy where the top levels are the most general description of function or process and the children are subset of the parents. Hierarchical structure complicates the inference of interactions. If a top-level GO term is used for prediction, then most proteins will be grouped. If a low-level GO term is used, then some proteins will not be grouped even when they share a similar biological process, but with slightly different tasks. A challenge is to discover the correct level of GO term to use and to exploit the parent – child nature of the hierarchy.

In one study, it was found that 76% of interacting proteins with known localization annotations shared common cellular compartments [51]. This information is the basis for the understanding that if two proteins do not localize to the same cellular component, they are unlikely to interact or dock. Like functional annotation, this process is inaccurate when used to predict interaction between proteins with similar component annotation because even when proteins localize to the same cellular component, that is no reason to believe they interact. *In silico* localization data can be used to enhance predicted interactions and docking, but not give information about the nature of the interactions or the specifics of the docking protein pairs.

Several other strategies besides localization information have been used to infer protein interactions. One such study [52] uses chromosomal proximity to infer functional coupling of proteins. Like localization, proximity of two genes does not provide enough evidence to make prediction, however it does offer additional evidence.

The term domain is loosely defined, by Jothi *et al.* [53] as a small region or sequence signature of the protein that mediates an interaction. Domain annotations can be determined through several experimental methods including x-ray crystallography and co-immunoprecipitation. Domain-domain docking prediction between proteins can be achieved through discovery of known binding domains on separate proteins and assuming that they dock with each other. This is true within a single species and between species. Other domain types like activation sites can be used to define protein–protein interactions. This method will have many false positives because of the large number of sequence signatures.

2.6 Prediction of protein annotation using sequence features

Although gene and protein annotation, like localization, domain, chromosomal proximity and gene expression generally are not able to predict protein interactions in isolation, they have been shown to be helpful when verifying protein interactions. Gene and protein annotation can also aid other research tasks like drug discovery and delivery. My early proteomics work inferred protein localization annotation given only the protein's amino acid sequence. The work [54] developed a hybrid classifier that uses similarity searching and a modified version of error correcting output code (ECOC) to classify proteins from four different protein datasets. We later extended this work (ModECOC) to handle multi-label prediction [55]. This modification was an improvement on single location predictive methods. We also used similarity searching to classify proteins when a similar homologous protein was discovered. Our method had many advantages over other multi-location predictors. In our algorithm when each binary base classifier voted a score was derived and the scores taken collectively gave weight to particular predictions.

Table 2.1 Confidence measures for YBR072W (a yeast protein)

Site	Confidence in this prediction (score from ModECOC)	Number of standard deviations from mean confidence	Site	Confidence in this prediction (score from ModECOC)	Number of standard deviations from mean confidence
actin	0.039295	0.9378	lipid particle	0.044667	0.1198
bud	0.03846	1.0649	microtubule	0.042375	0.4688
bud neck	0.039459	0.9128	mitochondrion	0.040632	0.7342
cell periphery	0.047322	0.2843	nuclear periphery	0.040391	0.7709
cytoplasm	0.064682	2.9274	nucleolus	0.042097	0.5111
early Golgi	0.045366	0.0135	nucleus	0.062616	2.6128
endosome	0.043408	0.3115	peroxisome	0.04347	0.3020
ER	0.043771	0.2562	punctate composite	0.047266	0.2757
ER to Golgi	0.043085	0.3607	spindle pole	0.04312	0.3554
Golgi	0.045979	0.0798	vacuolar membrane	0.048222	0.4212
late Golgi	0.048901	0.5248	vacuole	0.045416	0.0058

Table 2.1 shows an example prediction for a yeast protein. It is easy to see that the nucleus and cytoplasm classes are quite different as measured by the number of standard deviations their confidence score were from the mean of all confidence scores. These scores exceed a predefined threshold and the two locations should be predicted. This weight can be a more meaningful representation of the likelihood ratio instead of relying on a dataset like HPRD

as some sort of gold standard. Our algorithm also proved to have better class coverage which led to a higher overall recall and precision averages as seen in Table 2.2, which compares ModECOC against WolfPsort, another multi-class prediction algorithm.

Table 2.2 Some entries from ModECOC and WolfPsort confusion matrix for the plant dataset

	WolfPsort		ModECOC		ModECOC with Blast	
	recall	precision	recall	precision	recall	precision
nucleus	83.34	71.47	75.98	68.75	82.13	78.48
extracellular	92.14	76.33	82.14	84.56	92.14	90.85
vacuolar	0.0	0.0	21.74	62.50	47.83	73.33
cytoskeleton	0.0	0.0	85.29	93.55	85.29	90.63
Overall Average	33.35	39.33	35.24	47.11	48.92	57.95

Most algorithms, like WolfPsort, focus on the majority class, but ModECOC gives focus to all of the classes. Our work also showed that ModECOC handled many species without any change in parameters. WolfPsort reports optimal scores for each of the three types of species on which they applied their algorithm. Our algorithm performed well on the three datasets without setting parameters specific to each individual dataset. This shows our algorithm works without overfitting to a specific dataset making it applicable to a variety of datasets. A final advantage of our method over the WolfPsort method was the ability to change the combination and number of locations to predict. Besides WolfPsort we also compared ModECOC against PLPD [56], reliable protein localization prediction from imbalanced and overlapped datasets. We found WolfPsort only predicted from known pairs of locations and PLPD always predicted the same number of locations. Specifically, the WolfPsort animal dataset does not include any cyto_cysk proteins so the classifier will never predict this combination and therefore will never get this prediction correct. Because our classifier does not limit its predictions to the set of known predictions it can potentially get the prediction correct, making it more robust to new protein location predictions. We know that during cross validation with a known dataset our method will not get the unique predictions correct, but even with this built in reduction of accuracy, the method provides useful prediction. This was reinforced when a search of NCBI revealed several human proteins that localized to both the cytoplasm and the cytoskeleton. PLPD over-predicted, but considered the prediction correct if the actual class was among all of the predictions for a given test instance. This method leads to high recall, but low precision. ModECOC uses the confidence measures to define how many locations to predict, not a preset number of locations.

The ModECOC layered classifier was used in another project [57] that discovers transcription factors. In this paper we first identify only proteins that are nucleus bound, thereby limiting the number of genes that need to be analyzed via gene expression data.

I also found the Blast part of ModECOC that I developed for homology searching to be an efficient tool for discovering interesting information from within gene sequences. Specifically, Dr. Alex Bishop was researching the off-target effects of gene knock-down using double stranded RNA (dsRNA). CAN repeats, sequential repeats of any number of CAX nucleotides, are very common in nucleotide sequences. Because these patterns can be found in multiple locations, they should be avoided when designing dsRNA, to prevent off target effects. I implemented a method for finding the repeats, clustering them and ranking them according to their frequency.

Finally, I have also used homology searching in conjunction with research by Dr. Luis Penalva that identifies putative regulatory elements on un-translated regions (UTRs) of human mRNAs using an overrepresentation method. I used my homology tool to find overlaps between experimental and predicted binding sites.

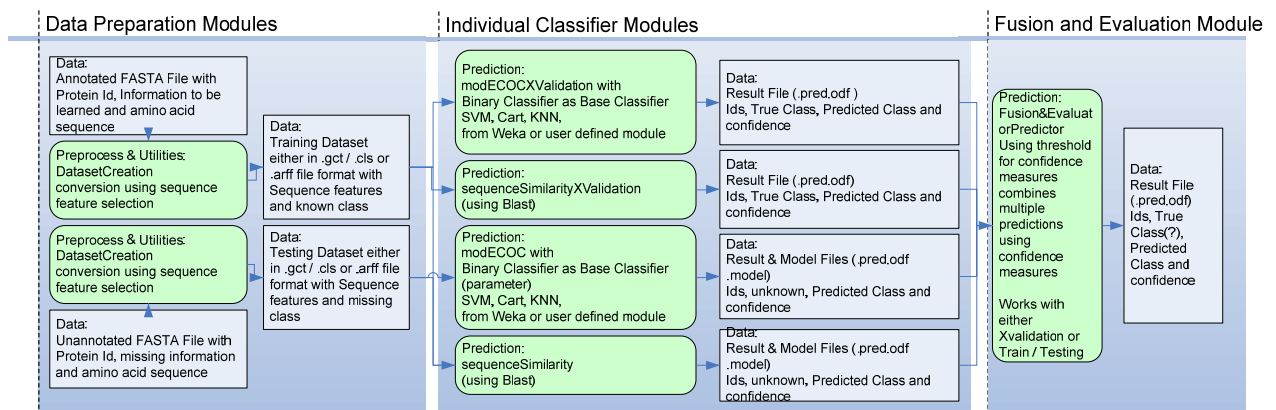


Figure 2.2 MiPPS GenePattern workflow

2.7 Finding protein annotations using GenePattern

My work involving protein annotation with various collaborators illustrated the difficulties with pure computational methods of analysis. ModECOC was programmed in Java and utilized BLAST [58] and several classes from the Weka [59] machine learning package. The data needed to be correctly formatted and the output processed. Generally speaking my collaborators were unable to use the program themselves and required my help for the protein annotation stage of their work.

To enable accessibility to ModECOC and homologous searching for prediction for other research; we combined them into a publicly available GenePattern [29] workflow. We implemented a sequence based multi-label, multi-class prediction workflow called MiPPS. MiPPS takes a two-pronged approach for protein classification by combining the results of a machine learning algorithm, ModECOC, and sequence similarity search based algorithm. Our approach was originally developed for subcellular localization prediction, however it is broadly applicable to other protein prediction problems including function, secondary structure and docking domains.

The MiPPS workflow, shown in Figure 2.2, has been developed for use in GenePattern's analysis environment. Users familiar with GenePattern's graphical approach of pipelining separate modules into a single tool will find the workflow simple to use. MiPPS includes data preparation, classification and fusion modules. The remainder of this section describes the modules I developed to implement MiPPS. All of these modules met the GenePattern rigorous quality-control standard and are published on the GenePattern website.

2.7.1 Preprocessing Modules: Arff2Gct, Gct2Arff, ProteinDatasetCreation

Two main approaches have been used to predict annotation information from a protein's amino acid sequence. Exploiting the belief that very similar proteins behave in similar ways allows the use of BLAST [58] to find a protein that is homologous to the unknown protein and use its class for the prediction. In this case the only feature to be used is the sequence itself. Another approach is to use machine learning techniques to infer a predictive model from proteins with known localizations [54]. This approach requires the conversion of the amino acid sequence to indices that characterize the biochemical properties of the protein. These biochemical properties [60] can take many forms from simple sequence counts of each of the amino acids to more complex features using pairs of amino acids [61]. Finally because using pairs of amino acids can cause an explosion in the dataset's features ($20 \times 20 = 400$ features), a reduced alphabet [62] can be utilized to group amino acids by their physicochemical properties like hydrophobicity.

The dataset creation module of MiPPS inputs a FASTA file that contains the known proteins, the class to be learned and their sequences. It provides the user with a list of features that can be selected: the sequence length (1 feature) amino acid composition (20 features), the

reduced alphabet composition of the n (19 features) and c (19 features) terminal regions, and the gapped pairs from the reduced alphabet with a selection of how long the gap between amino acids is (55 features for each of the gaps). The module then outputs the dataset in either a GenePattern file format (combination of .gct and .cls) or as a .arff file used by Weka [59] and BioWeka [63] classifiers. By choosing the features, researchers can fine tune the classifier to their type of classification.

2.7.2 Prediction Modules: ModECOCTrainTest and ModECOCXValidation

Most machine learning classifiers provide single class output from a set of possible outputs or multi-class learning by making multi-classes their own distinct class. In the context of subcellular localization prediction this would mean a single class protein would be classified as possibly localizing to the cytoplasm or the nucleus, but not both. For multi-class learning some proteins would be labeled as cytoplasm, some proteins would be labeled as nucleus and a different set of proteins would be labeled as cytoplasm/nucleus. When a classifier then learns from this data and is used for prediction, an unknown protein would be labeled from this set of three classes.

For some datasets this approach results in a huge number of classes. For instance in the Huh *et al.* [64] yeast dataset, there are 22 different subcellular compartments, but 95 different combinations of single and multi-locations represented by the proteins. Traditional machine learning algorithms like SVM, K-nearest neighbor or Error Correcting Output Code (ECOC) will have a difficult time discriminating between all of the possible classes. Also, if an unknown protein is predicted that does not locate to one of these 95 classes, it has no chance at being predicted correctly, because the classifier only chooses from the classes it has learned. ModECOC developed by Doderer *et al.* [55] addresses the need for a classifier that could handle multi-classes with the ability to make *a priori* predictions of combinations of classes not seen in the training set.

I implemented ModECOC as two GenePattern modules: ModECOCXValidation allows for leave-one-out testing and ModECOCTrainTest, which uses training and testing datasets for evaluation of the accuracy of the classifier. The result of using the training and testing datasets will be prediction of the testing dataset proteins by the training set and a confidence measure for the prediction.

2.7.3 Prediction Modules: BlastTrainTest and BlastXValidation

Using the same test and training set as ModECOCTrainTest and ModECOCXValidation, this module first uses Blast's utility program, formatDB, to create a searchable database of known proteins. Then it uses BlastAll to search for homologous proteins for each of the test proteins. The output will be the test dataset with the predictions and the confidence measure.

2.7.4 Utility Module: CombineOdf

The input for this module will be the predictions from the ModECOC and Blast modules. In a hybrid approach, the classification that is assumed to be more reliable is used for the final classification of an unknown protein. A homologous protein, if it exists, can offer very good predictive accuracy. On the other hand, if the score for the sequence similarity is too low, homologous searching has very poor accuracy. So, if the input parameter threshold is met then the label for the most homologous protein will be used to annotate the unknown protein. Otherwise, the label from ModECOC will be used. If both methods agree on the prediction, this label appears quite valid. The output from this module provides the predictions and confidence measures for the prediction.

2.7.5 Discussion

The MiPPS suite of tools has several benefits for researchers. First using the available datasets, protein location prediction can be made for proteins of interest that have not already been annotated. Secondly, the tedious conversion of FASTA files into a format to be used in classifiers is encapsulated in a GenePattern module. This together with several useful feature selections can be used in other types of classifiers like SVM and neural nets. The ModECOC algorithm has been shown to be an effective multi-class prediction algorithm and can be used in other prediction problems. Finally, the module that combines two prediction outputs can be used here for similarity searching and ModECOC, but can also be used to combine predictors for other suites.

GenePattern allows analysis steps to be combined together and by using all of the modules the initial multi-location prediction work can be automated, however this workflow is not easily integrated with the other workflows and suites of tools provided by GenePattern. For example, if the goal of protein localization annotation was to increase belief in protein

interaction prediction or existing interaction data, the annotation and interaction data is not easily joined using GenePattern.

2.8 Evaluation of public domain protein-protein interaction data

Many previous predictive techniques have used networks to represent the protein-protein interactions. Creation of the networks is accomplished by simply analyzing a single data source for potential interactions and filtering out the interactions that are deemed false positives. The choice of types of data sources is quite varied as described earlier, and each choice has strengths and weaknesses for discovering protein interactions. For example, if only localization to the same component as indicated by GO component similarity is used to infer interaction, then the resulting interaction network will have high coverage with low confidence.

Another choice for creation of interaction networks is to use multiple sources of data. This, theoretically, will fill in the gaps when one dataset misses some of the interactions. The databases in this survey contain the interaction networks that were created from the single or multiple sources of experimental data. Usually they are added to by either biologists who have found experimental interactions in literature or by investigators who have determined new interactions by some experiment they have carried out. The data in the dataset then takes on the qualities of the source of data. For example, if the database is created through mainly mass spectrometry experiments, the data base will contain interactions that constitute complexes, but not transient interaction partners. If someone is interested in new protein-protein interaction techniques based on mass spectrometry data, they need only use a well known database that has mass spectrometry data. At the same time a biologist interested in a particular protein can access the database for actual predictions about that protein's interacting partners. The proliferation of publicly available biological data sources greatly increases potential for discovery, but also raises issues of complexity, organization, and reliability for the user. Multiple data sources increase coverage and confidence in the annotated information, but sources are quite varied in their reliability and format, each with strengths and weaknesses.

Mathivanan *et al.* [27] evaluated several human protein-protein interaction databases that are available for download to aid researchers in choosing the data for their experiments. The databases included BIND [48], DIP [65], HPRD [66], IntAct [46], MINT [67], MIPS [68], PDZBase [69], and Reactome [70]. They discovered significant variations among these databases

in terms of number of genes represented, number of interactions, source of data for the databases, and vocabularies. The source of databases could be high-throughput experiments (experimental), automatic text-mining of articles (automated) or computational predictions (predicted). Their comparisons are shown in Table 2.3. They also showed the overlap between datasets in terms of the proteins and the interactions found. This is shown in Figure 2.3.

Table 2.3 Unique features of human PPI databases

Features Databases	Human PPIs	Protein count	PPI curation methodology	Unique features
BIND	6,621	3,887	Experimental	Protein complexes, biological pathways, non-protein interactions, Data for >1473 organisms
DIP	1,067	804	Experimental	PPIs for other organisms, protein complexes
HPRD	36,617	9,427	Experimental	Protein annotations are included (e.g. PTMs, substrate information, tissue expression, disease association, protein complexes, subcellular localization). Signal transduction pathways
IntAct	10,244	4,614	Experimental	Protein complexes, PPIs for other organisms, non-protein interactions, provides web based applications, ProViz and Hierarch View, for visualization of interactions
MINT	11,367	4,975	Experimental	PPIs for other organisms, non-protein interactions
MIPS	346	405	Experimental	PPIs for other organisms
PDZBase	101	115	Experimental	PPIs involving PDZ domains. Prediction of residues that interact.
REACTOME	5,960	970	Experimental, automated and predicted	Biological pathways for several organisms. Navigation through reactions in biological pathways and visualizing connections between them

(A)

HPRD (36,617)								
BIND (6,621)	4,903							
DIP (1,067)	801	264						
MINT (11,367)	8,690	1463	379					
Reactome (5,960)	538	207	67	102				
IntAct (10,244)	8,031	1167	283	7,362	173			
MIPS (346)	307	294	28	65	14	43		
PDZ Base (101)	93	19	0	60	0	5	3	
	HPRD (36,617)	BIND (6,621)	DIP (1,067)	MINT (11,367)	Reactome (5,960)	IntAct (10,244)	MIPS (346)	PDZ Base (101)

(B)

HPRD (9,427)								
BIND (3,887)	3,414							
DIP (804)	755	537						
MINT (4,975)	4,719	2218	562					
Reactome (970)	733	453	164	497				
IntAct (4,614)	4,421	1969	473	3795	497			
MIPS (405)	396	390	146	303	78	262		
PDZ Base (115)	114	64	10	99	1	54	16	
	HPRD (9,427)	BIND (3887)	DIP (804)	MINT (4,975)	Reactome (970)	IntAct (4,614)	MIPS (405)	PDZ Base (115)

Figure 2.3 Overlap of PPIs and proteins in human PPI databases.

(A) Pairwise overlap of protein interactions across databases is shown in cells. The number of non-redundant direct PPIs present in each database is shown in parentheses for each database. (B) Pairwise overlap of proteins across databases is shown in the cells. The number of non-redundant proteins present in each database is shown in parenthesis for each database.

The main idea as presented were the number of proteins and their interactions varied greatly across databases as can be seen in Figure 2.3. Further they note that there exists very little overlap in the actual interactions annotated. So, taken individually one dataset could not be relied on for full coverage of the possible interactions. Mathivanan *et al.* attributed this to the difficulties that a non-standard vocabulary can bring. Since each biologist annotate their experimental results with slightly different interpretations. They concluded a standardized vocabulary is necessary, but not currently enforced.

Integration of protein interaction databases would provide wider coverage of a species' actual interactome; however care must be taken when comparing protein identifiers. NCIBI's MiMI [12] integrates these by compiling an extensive number of databases to increase coverage of many types of information while rectifying vocabulary differences. Many of the data sources covered by Mathivanan are present in MiMI. Another important step involves using protein annotation to increase confidence in the protein interactions present in the databases.

OPHID [71] is one example of a database that incorporates protein annotations like protein domains, gene co-expression and Gene Ontology terms to evaluate and increase confidence in interactions present in its database. It also allows users to search its database through a web interface. String [15] probably contains the most comprehensive source of information on protein interactions. At the time of its most recent publication it contained about 2.5 million proteins from 630 organisms. String incorporates multiple sources of interactions and uses a variety of protein annotation for confidence measurements describing each interaction. String also provides access to the database information through a URL-based programming interface. This extensive repository of information on protein interaction can enable a scientist to find information ranging from few, well annotated, very high confidence interactions to many, low confidence interactions.

If a biologist is interested in research like pathway, drug discovery and drug discovery, they could benefit from interaction information and can call String for protein information. However, the biologist would need to form the URL query and parse the return from String either manually or by creating a computation algorithm to do so. Integrating the information with an existing analysis could also prove difficult.

CHAPTER THREE: SIDEKICK

ModECOC, InterologFinder, and the GenePattern MiPPS workflow are all meaningful tools that expand the information available to biologists. ModECOC enabled *a priori* prediction of protein annotation, and InterologFinder combined known and predicted protein interactions and provided confidence measurements for each. MiPPS placed predictive and utility algorithms in an application accessible to any scientist familiar with GenePattern's modules. However these tools lack an integrated approach to a variety of analyses using diverse and up-to-date data. Also, when scientists disagreed on how to measure confidence, as I saw during the development of InterologFinder, one method was chosen over others and implemented leaving only one opinion and disregarding the others. Sidekick was conceived and developed to enable biological discovery including, but not limited to, protein interaction discovery while addressing these issues.

3.1 Overall organization and purpose

Sidekick is a web-based biological knowledge discovery application that focuses on bottom-up discovery and organization of belief. Sidekick combines multiple sources of data for many common research tasks including determination of genes involved in a disease, diseases associated with a gene, gene expression enrichment, Gene Ontology enrichment, chromosome locality enrichment, and interactions. By using web services, Sidekick keeps its information as current as the data sources themselves. The user can save and combine analysis steps in order to easily document and reproduce results or back track to previous states when investigations in one direction do not produce meaningful results.

Currently Sidekick supports three queries, four filters, several ways to combine results, and methods for saving and restoring workflows and data. Sidekick's modular design using Adobe Flex and ActionScript 3 allows programmers to incorporate additional queries, filters, or data sources. Sidekick runs in any browser with the latest Adobe Flash player plug-in.

Sidekick has a unique system for managing user belief that makes the user an active participant in assigning confidence measurements to biological discoveries. Users can combine various quality measures provided by different sources to evaluate quality of the analysis. Furthermore, users can incorporate their own biases related to the specific sources of information

language developed by Adobe and a XML based mark-up language called MXML to create Flex projects. ActionScript is a JavaScript-based language that follows a Java-like objected-oriented paradigm and provides the ability to handle complex algorithm creation. A goal of ActionScript/Flex is to enable the development of platform-independent web applications that have the feel of desktop applications by providing a communication infrastructure and an AJAX-enabled GUI widget kit. The underlying AJAX infrastructure allows the browser to respond to user events such as mouse clicks locally without refreshing the web page. MXML is an XML-like mark-up language included as part of Flex that allows users to specify the application graphical layout as a text file. The use of both ActionScript and MXML is similar to web site creation where a program will use both html and JavaScript. MXML specifies the layout and the ActionScript provides the functionality. The MXML file also can contain ActionScript code and instantiate and use objects defined by ActionScript classes. The Flash movie (.SWF) that is generated when the IDE compiles the project is saved to a web server. It can be downloaded and run in any browser with the latest Adobe Flash player installed. The ActionScript is interpreted by the Flash player plug-in rather than directly by the browser.

The Adobe Flex technology that drives the IDE, SWF movie product and Flash Player is designed to be browser-independent. Users should be able to work with Sidekick regardless of which browser or which operating system they are using. Sidekick runs on the user's system enabling many simultaneous users without concern for competition for service resources, which would happen on a system where the code runs on the server. Finally, Adobe Flex is a good platform for Sidekick's development because of its rich web service query interface that allows for asynchronous communication between Sidekick and its data providers, producing a "desk top application" experience in a web application.

3.3 Sidekick uses up-to-date information

Sidekick uses information from core resource sites such as NCBI, NCIBI, and EBI, to get the most recent and best-quality available information. Table 3.1 summarizes some of the major sites used by Sidekick and the information they provide.

For frequently used information or when web services are not available, local web services have been developed that run on the server where Sidekick resides (<http://visual.cs.utsa.edu>) and manage scheduled downloading of up-to-date data. Also, due to

security policies of Adobe Flash applications and overuse policies of public data sources, a server side portal was created to allow and optimize communication with the resource sites. This portal was implemented using the SideCache web service communication suite, jointly developed by myself and other in the UTSA Visualization Laboratory.

Two types of server side web services make up the SideCache web service communication suite. The first, WebServiceProxy, enables communication between a program running on the user's computer and the websites that expose their information with web services. Due to security risks, Adobe Flash prevents programs from calling web services directly. The WebServiceProxy service, written in Java and runs in Java, is able to call web services. At the most simple level, WebServiceProxy acts as a pass through enabling communication between client and data sources. However, WebServiceProxy also maintains a cache of queries and results that is exposed to all users of Sidekick. WebServiceProxy also manages timing of web service calls. To prevent overuse by individuals, some web sites including NCBI define maximum number and frequency of queries by users. If the maximum is abused, the user is prevented from future queries being fulfilled. Because queries from multiple users all come into WebServiceProxy, it maintains a queue of pending requests and executes them in accordance with the guidelines set up by the data provider.

The second form of SideCache server side support manages queries that require data from web sites that do not expose information via web services, but do allow file downloads. For example, Ensembl does not allow direct queries for orthologous information for genes, however it does allow downloading a file of all orthologs between two species. To expose this information to Sidekick a server side servlet was written in Java. First, the files with the information are downloaded from the public website are stored on the server. The files are automatically updated at off-times according to a pre-specified schedule based on the file update schedules of the source websites. The schedule is managed by SideCache. Second, the server side servlet actively waits for requests from Sidekick. When a request from Sidekick arrives into the server side servlet the information requested is determined from the downloaded information, written into XML format and returned to Sidekick. In figure 3.2 the SideCache – Data Sources communication occurs only when the data needs to be refreshed, however the Client – SideCache communication occurs each time Sidekick requests information from a server side servlet.

Table 3.1 Data providers and their benefits

	<p>NCBI – National Center for Biotechnology Information (http://www.ncbi.nlm.nih.gov/) has compiled perhaps the most comprehensive gene database including gene names, their common identifiers, chromosome and chromosome starting position, interactions the genes are involved in and annotations like their GO ontology terms.</p>
	<p>Ensembl (http://www.ensembl.org) maintains the orthologous relationships between many genes across multiple species. They also maintain the identity score between each orthologous pair.</p>
	<p>NCIBI's MiMI (Michigan Molecular Interactions) (http://mimi.ncbi.org) compiles several publicly available data sources including:</p> <ul style="list-style-type: none"> • BIND: Biomolecular Interaction Network Database • BioGRID: Biological General Repos. for Inter. Data. • CCSB: Center for Cancer Sys. Biology Inter. Data. • DIP: Database of Interacting Proteins • HPRD: Human Protein Reference Database • IntAct: Molecular Interaction Database • MDC: Unified Human Interactome • MINT: Molecular Interaction Database • Reactome: Curated Knowledgebase of Bio. Path.
	<p>NCIBI – the National Center for Integrative Biomedical Informatics (http://www.ncibi.org). NCIBI's Gene2MeSH uses a statistical approach to annotate genes reliably and automatically with the concepts defined in MeSH, the National Library of Medicine's controlled vocabulary for biology and medicine.</p>
	<p>European Bioinformatics Institute (EBI)'s web service Gene Expression Atlas (http://www.ebi.ac.uk/microarray-as/ae/), within ArrayExpress for calculation of enrichment based on gene expression. ArrayExpress provides a large corpus of microarray data sets that have been hand-curated and labeled with attributes such as cell type, developmental stage, and disease state among many others.</p>
	<p>The hierarchy of MeSH terms is found from the National Library of Medicine's controlled vocabulary thesaurus of Medical Subject Headings (http://www.nlm.nih.gov/mesh/meshhome.html).</p>

3.4 Tasks

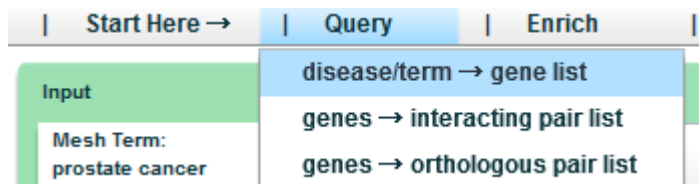


Figure 3.2 Query screenshot

Sidekick implements many features needed for biological research. These include pair analysis similar to those in InterologFinder and annotation of individual entities. But Sidekick goes beyond those tasks with added features including flexible combination of evidence, enrichment of subsets of items to find commonalities, and belief management. These common research tasks are available to scientists with simple mouse clicks, drop-down selections, and text field fill in for search terms. Complex combinations of tasks can be visualized as a set that displays ordering of tasks completed, however the input of a step is any result, not necessarily the one adjacent in the workspace. Sidekick workflows often begin with queries to generate gene lists or gene-pair lists as illustrated in Sidekick's query menu of Figure 3.2.

3.4.1 Getting a gene list from a search

The Query menu, *Query: disease/term → gene list* generates a list of genes given a search term by combining results from both NCBI and NCIBI's Gene2Mesh service.

The Gene2MeSH web service returns gene symbols given a MeSH term or MeSH terms for a given gene along with p-values indicating confidence in the results. The NCBI ESummary web service produces a gene list given a term (not necessarily a disease term). Users can choose between *disease only* and *general* within the NCBI search input filter to focus on only disease terms or to allow for generalized searches. NCBI does not provide a confidence value.

By combining both web services, Sidekick provides greater coverage of genes given a MeSH term. The general / disease only setting in NCBI is one example where the research task is driven by user belief and expectations. Take for example a search for a very specific disease like Alzheimer's. Using disease only will return a focused set of genes, however a user who is interested in a specific gene like the MED26 gene, can search for that term with general selected to get the gene.

The initial call:

a) `http://eutils.ncbi.nlm.nih.gov/entrez/eutils/esearch.fcgi?db=gene` for NCBI

b) `http://gene2MeSH.ncibi.org/fetch.php` for NCIBI

go through the SideCache system addressed by:

c) `http://visual.cs.utsa.edu/WebServiceProxy/proxy/?url=`

and return gene IDs or gene names, respectively. After the information is returned from the web service calls, Sidekick makes additional calls to complete each gene's profile (gene ID, gene name, chromosome, chromosome starting position and interactants). Because this information is accessed so frequently, a server side servlet, GeneName2idWS, was developed. Managed by SideCache, it downloads several files from NCBI that are parsed, combined and loaded into a synchronized map for safe O(1) lookups. This methodology meets the requirement of up-to-date data, which is safe for many users while minimizing the time it takes for web service calls to NCBI directly. The files downloaded are:

a) `ftp://ftp.ncbi.nlm.nih.gov/gene/GeneRIF/interactions.gz`

b) `ftp://ftp.ncbi.nlm.nih.gov/gene/DATA/GENE_INFO/Mammalia/Homo_sapiens.gene_info.gz`

c) `ftp://ftp.ncbi.nlm.nih.gov/genomes/MapView/Homo_sapiens/sequence/current/updates/seq_gene.md.gz`

for each species (the human example is shown).

3.4.2 Producing gene-pair list

Pair lists, as seen in InterologFinder, convey relationships such as interactions and remain an integral part of genomic research. Tools for pair-wise analysis are necessary for interaction research, but pair lists are not limited to only interaction relationships. Furthermore, pair information is often not the end of the analysis process, but rather an intermediate step for many types of research questions. Sidekick's flexible approach enables better use of this information. The case study of Chapter 6 utilizes interacting and orthologous pair lists to enable the complex creation of a predicted interaction network with just a few mouse clicks. Complex relationships are easily visualized and combined.

Queries such as *Query: genes* \rightarrow *interacting pair list* and *Query: genes* \rightarrow *orthologous pair list* generate and display relationships between genes and enable gene-pair analysis.

The first, *Query: genes → interacting pair list*, searches several data sources for genes believed to interact with the input gene list. The web services that provide the interaction data are NCBI and NCIBI's MiMI [12]. MiMI was created by compiling several publicly available data sources including HPRD [66], IntAct [46], BIND [48], BioGRID [72], MINT [67], CCSB [73], DIP [65], Reactome [70], and MDC [74]. Sidekick displays the input genes in the first column and the corresponding output gene interactants in the second column of the Sidekick Results window. As is the case of a single gene list, multiple data sources allow for better coverage of the possible interactions and increase confidence when results are found from more than one source.

The second query, *Query: genes → orthologous pair list*, finds orthologs of genes between species. This query is applied to move from one species to another and back again. Once the users have gene lists, they can find orthologs using Sidekick and use these pair lists to draw conclusion in one species from data available in another species.

The user inputs a gene list and selects a target species from the other four species currently supported. Sidekick uses Ensembl's orthologous gene lists. The output genes are orthologs corresponding to the input gene.

For *Query: genes → interacting pair list* the NCBI information is handled by the GeneName2idWS servlet using the data downloaded as described in section 3.4.1. MiMI's web service call goes through SideCache's WebServiceProxy and takes the form:

`http://mimi.ncibi.org/MimiWeb/interaction-list-page.jsp?geneid=`

The *Query: genes → orthologous pair list* uses a server side servlet called OrthologizerWS because Ensembl does not expose its data via web services. OrthologizerWS schedules automatic downloads of data from Ensembl using SideCache. It parses the results, and forms species pair files after matching identifiers. This data is exposed via URLs of the following form:

`http://visual.cs.utsa.edu/OrthologizerWS?sourceSpecies=9606&targetSpecies=10090&genes=`

Orthologizer returns the IDs of the orthologous genes for each input gene after looking through the appropriate ortholog file. The files required are found at the Biomart area of Ensembl website `http://www.biomart.org/biomart/martservice`. For each species we download a thesaurus file that matches Ensembl IDs to NCBI IDs and for each pair of species (e.g. 9606/10090 or

10090/9606) we download the orthologs and their percent identity. Each is identified using its Ensembl ID.

3.4.3 Supporting information for results

Sidekick also displays the sources of information for all results. If multiple web services returned the same result, Sidekick lists each source. The user is able to reflect their belief in the results by assigning confidence to the evidence that supports the gene or gene-pair lists. Sidekick's Belief management is detailed in Chapter 5.

3.4.4 Displaying results

Gene id (Name)	Score-from-source (Raw P-Value)	Score-from-source credibility	Source
675(BRCA2)	25(0.00e-16)	0.5	NCBI;NCI
8068(BRCATA)	1.301(5.00e-2)	0.5	NCBI
2263(FGFR2)	7.44(3.63e-8)	0.5	NCBI;NCI
27324(TOX3)	9.993(1.01e-10)	0.5	NCBI;NCI
146253(LOC1462...)	1.301(5.00e-2)	0.5	NCBI
8438(RAD54L)	1.301(5.00e-2)	0.5	NCBI
672(BRCA1)	25(0.00e-16)	0.5	NCBI;NCI

Figure 3.3 gene list screenshot

One reason Adobe Flex was a natural platform on which to build Sidekick was its collection of graphical objects, one of which is a display grid (DataGrid - seen on right side of Figure 3.3) that is richly functional and highly flexible. The DataGrid is used to display the results for all of the queries. Each type of query follows a general pattern for the columns to provide continuity between queries and reuse of coding. Generally speaking, the gene results appear in the first column for single lists or in the first two columns for gene-pair lists. The identifiers are hyperlinked to the NCBI website for easy look-ups of gene information. This functionality was not built-in, however the flexibility of the grid object allowed for additional functionality to be added. The remaining columns display supporting information and credibility scores for the gene results. For standard data types, the columns are sortable. However, user-defined sort functions can be applied to a column for user defined data. Adobe Flex DataGrid objects are simple to configure. Because Sidekick was developed in a consistent manner, the

programmer simply needs to copy and paste the column and grid code from a similar query to replicate the grid when new modules are added.

3.4.5 Workspace of results

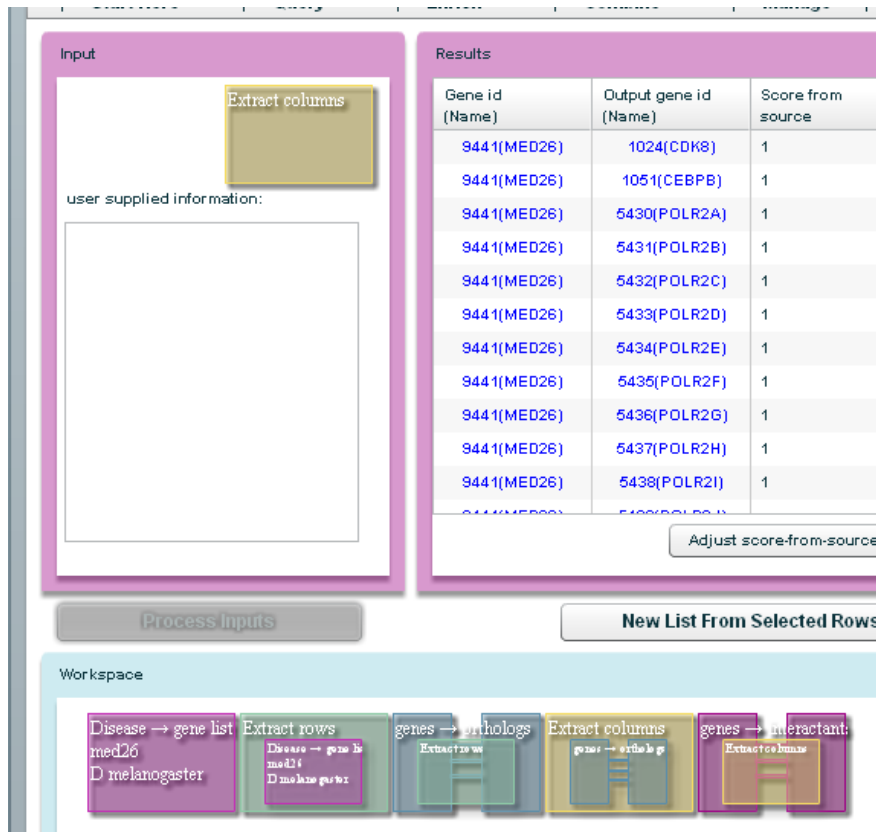


Figure 3.4 Sample Sidekick workspace



Figure 3.5 Icon screenshot

The workspace (see the bottom of Figure 3.4) provides a visual connection between analysis steps. Sidekick represents each gene list or a gene-pair list in the workspace by an icon. The colors of an active query or enrichment icon match the frame of the Results to indicate the connection between the current step and its results.

In the close-up of two result icons shown in Figure 3.5, the left rectangular icon represents the gene list resulting from searching for *breast cancer* in humans. The second double rectangular icon represents the gene-pair list resulting from determining orthologies of the first

list in mouse. The nested inner rectangle in the second icon designates, both by color and text, that the first list was its input. Results that were produced from an input have a small copy of the input icon to indicate flow of data. The icons are clickable to allow the user to easily jump among results.

Unlike many analysis environments, Sidekick does not attempt to manage workflows as directed graphs driven by execution order of steps. Automatic workflows, such as those seen in GenePattern can be helpful if a scientist wants to repeat an analysis using the same steps but with different data. This is not what Sidekick was designed for. The discovery process is rarely accomplished in a linear progression. Sidekick organizes the workspace around individual icons that show their input relationships. The user manages the progression of discovery by clicking a result and using it for other steps. This means that one result could be the input for many other steps and what is described by the icons is the relationship between successive pairs of steps, not the sequential order of all steps.

Each result also has an area for the user to supply information (see the left side of Figure 3.4). This area could be used for process annotation, highlighting important findings, or incorporating other notes about data sources. The user is also free to ignore this feature.

3.4.6 Extracting a new list from a Sidekick result

657(BMPR1A)	25(1.39e-166)	0.5	NCBI;NCBI
654(BMP6)	25(8.49e-147)	0.5	NCBI;NCBI
658(BMPR1B)	25(8.95e-125)	0.5	NCBI;NCBI
2099(ESR1)	25(7.47e-97)	0.5	NCBI;NCBI
1077(ESR1)			

Adjust score-from-source credibility Adjust source

New List From Selected Rows Ne

Figure 3.6 Extracting screenshot

The ability to extract data from results enables the user to focus on the information important to other research steps. This also mitigates large results that can occur for example, in the case of forming a gene-pair list through interactions.

To form a new list the user highlights the rows (see Figure 3.6) to include using *shift-click* rather than dragging. (Sidekick highlights the selected rows in blue.) For simple gene lists, the user presses the *New List From Selected Rows* button to create a new gene list. For gene-pair

lists, the user presses the *New List From Selected Rows* button to create a new gene-pair list. Alternatively the user can press the *New List From Result Column* button to create a gene list from the selected rows of the *Results* section *Output gene id* column. (The *Gene id* column already corresponds to another icon in the workspace.)

3.5 Enrichment

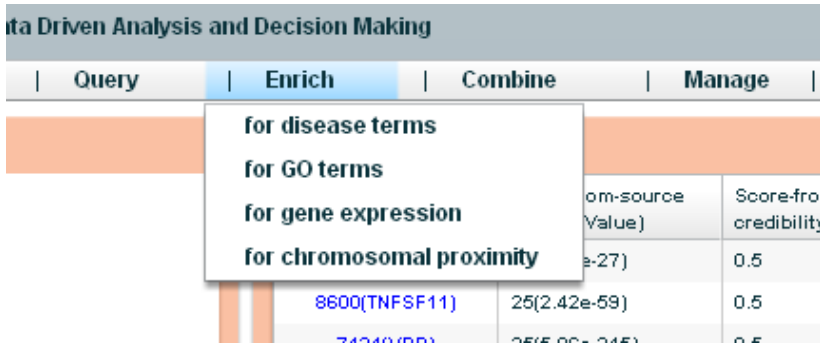


Figure 3.7 Enrichment screenshot

Enrichment looks for a common feature among elements of a study set that occurs more frequently than at random as determined by a population set. Sidekick's enrichment options shown in Figure 3.7 discover concepts that are enriched in a list of genes or in a list of gene-pairs. Sidekick currently supports enrichment by GO terms, disease terms, gene expression, and chromosomal proximity.

Profile	Gene id (Name)	Score-from-so...	Score-from-so...	Size of group	Gr...
▶ Congenital, Hereditary, and Neonatal Dis			0.5	19	0.5
▼ Endocrine System Diseases(7 of 24 gen			0.5	7	0.5
Endocrine System Diseases(7 of 24	1499(CTNNB1)	2.387(4.09e-3)	0.5	7	0.5
Endocrine System Diseases(7 of 24	7490(WT1)	2.387(4.09e-3)	0.5	7	0.5
Endocrine System Diseases(7 of 24	2719(GPC3)	2.387(4.09e-3)	0.5	7	0.5
Endocrine System Diseases(7 of 24	7849(PAX8)	2.387(4.09e-3)	0.5	7	0.5
Endocrine System Diseases(7 of 24	6736(SRY)	2.387(4.09e-3)	0.5	7	0.5
Endocrine System Diseases(7 of 24	3481(IGF2)	2.387(4.09e-3)	0.5	7	0.5
Endocrine System Diseases(7 of 24	675(BRCA2)	2.387(4.09e-3)	0.5	7	0.5
▶ Female Urogenital Diseases and Pregn			0.5	20	0.5
▶ Fetal Macrosomia(1 of 24 genes)			0.5	1	0.5

Figure 3.8 AdvancedDataGrid screenshot

Sidekick relies on another type of grid available in Flex called the AdvancedDataGrid (see Figure 3.8) for results returned from enrichment analyses. This grid, unlike the one used for

gene and gene-pair lists, allows the user to group results together based on an equivalent term. In the case of enrichment, the term is generally the concept found to be enriched for, however because of the flexibility of the AdvancedDataGrid, the grouping is easily changed, and allows the user to select *group by input genes* and *group by output genes* in the case of pair enrichment. This added flexibility, for example, allows the user to see all of the diseases related to a single gene.

3.5.1 Enrichment for GO terms

Enrich: genes for GO term enrichment explores the relationships among genes in a given a gene list according to their Gene Ontology (GO) annotations. This information could be important to scientists seeking to add genes to existing biological pathways. The inputs for this filter include the *GO type* (Component, Biological Process or Function), *Enrichment strategy*, *Evidence included*, and the *Maximum # of GO term hits* to return.

A traditional approach to GO term enrichment takes each GO term and determines if that term is over-represented within the gene study set, as compared to a larger set gene population. The terms in the Gene Ontology are not independent, but rather form a directed acyclic graph with more specific terms as the children of more general parents. For example, mismatch repair is a child of DNA repair. Simple term-for-term analysis does not take into account the potential relationships among different GO terms such as a parent-child relationship.

Grossmann *et al.* [75] present a novel approach for detecting overrepresentation of GO terms using parent-child analysis. Their method addresses not only the hierarchical nature of GO terms addressed but also occurrences of the same term in multiple branches of the graph. The less rigorous *Parent-Child-Union* strategy defines the set of parents of a term *t* in the population and study sets as the union of genes annotated with *any* parent of *t*. The *Parent-Child-Intersection* strategy reduces the number of enriched terms by defining the set of parents of a term *t* in the population and study set as the intersection of a term *t* in the population and study sets, respectively, counting the genes annotated to *all* of the parents of the respective sets. Grossmann *et al.* conclude the parent child approach avoids many of the false positives that the *Term-For-Term* approach produces.

IEA, Inferred from Electronic Annotation, consists only of evidence from computational analysis and is considered by some as less trustworthy. The *Evidence included* allows the user to

check either *Curated Only (no IEA)* or *All Types (include IEA)*. Selecting *All Types* increases the evidence, but perhaps decreases the perceived quality. The *Max # of GO term hits* allows for either targeted or more general searches.

The parent-child strategy was made available by Grossman *et al.* for non-commercial use through a program called Ontologizer. Ontologizer was written in Java, contains more than 250 classes, and provides the source code. Ontologizer expects the presence of three files including the general population (all genes and their GO annotations), the study set of genes, and the hierarchical list of GO terms. To utilize the Java application the code was incorporated into a server side servlet called GoEnrichWS, which was written in Java, and is managed by SideCache. The servlet exposes the enrichment task through a URL of the form:

```
http://visual.cs.utsa.edu/GoEnrichWS?enrichType=goEnrichment&evid=_IEA_&type=Component&calc=Parent-Child-Union&species=9606&max=25&genes=8929,429,1387
```

This example handles parent-child union, localization enrichment for the three human genes including all evidence and limiting the results to the 25 best terms: The files required by the Ontologizer code are the annotated lists of genes downloaded and unzipped from NCBI:

```
ftp://ftp.ncbi.nih.gov/gene/DATA/gene2go.gz
```

and the GO hierarchy from the Gene Ontology organization:

```
http://www.geneontology.org/ontology/obo_format_1_2/gene_ontology.1_2.obo.
```

These files have a scheduled download managed by SideCache.

One additional step was required for proper use of the population file. Because we need to compare our study set of genes for inclusion in a population, the population file must match the input terms including species, type and evidence. The NCBI file is separated into all combinations of these input terms after download, so that the web service calls require only file use and not preparation. There were 30 (5x3x2) files created for the 5 species, 3 GO types and 2 evidence types. After the input parameters are parsed, the GoEnrichWS servlet calls the method for the Ontologizer classes with the proper population, study set, and parameters. The GoEnrichWS servlet returns any results in XML format that Sidekick parses and uses for input into the AdvancedDataGrid.

3.5.2 Enrichment for disease terms

Enrich: for disease terms enrichment uses NCIBI's Gene2MeSH web service to find the disease MeSH terms enriched for subsets of the gene inputs. Sidekick only uses the disease

category MeSH terms from the NCBI's National Library of Medicine. Like GO terms, disease MeSH terms are hierarchical. Sidekick uses a modified version of the GO term enrichment algorithm of Grossmann *et al.* to combine MeSH terms in parent-children relationships. The inputs include *Enrichment strategy* and the *Max term hits*. The outputs are similar to those of GO term enrichment, with the *Score-from-source* as the p-value representing the likelihood that a subset with shared MeSH terms happened randomly as compared to the general population. A scientist might be able to use disease enrichment when looking for genes involved in complex diseases with multiple phenotypes.

Because the infrastructure for hierarchical enrichment was in place; using a strategy similar to that of GoEnrichWS was a natural choice for disease enrichment. An additional operation was added to the web service and the appropriate files were organized for scheduled download. Unlike GO enrichment, there is only one type of evidence and one type of term enriched for. A sample URL call for disease enrichment is:

```
http://visual.cs.utsa.edu/GoEnrichWS?enrichType=MeshEnrichment
&calc=Parent-Child-Union&species=9606&max=25&genes=8929,429,1387
```

The file downloads require more steps than the original Ontologizer. The hierarchy of MeSH terms is found from the National Library of Medicine's (NLM) controlled vocabulary thesaurus of Medical Subject Headings. This file is downloaded by hand and copied to the visual server. The NLM web site does not allow automatic downloads, however the MeSH terms are only updated annually, mitigating the burden of this hand processing step. Ontologizer was built to manage GO terms and so the MeSH terms are assigned unique GO terms and an OBO file is constructed that mimics the GO OBO file while maintaining the hierarchical nature of the MeSH terms.

During the file download step of GoEnrichWS, managed by SideCache, the population file is built by accessing NCIBI's gene2MeSH web service for each MeSH term in the NLM file. For example the URL:

```
http://gene2MeSH.ncibi.org/fetch.php?MeSH=" Brain Abscess"
```

finds all genes related to Brain Abscess. This step requires separate calls for each of the 4408 different MeSH terms. The genes and their corresponding MeSH terms represented by assigned GO terms are stored in species specific files.

During the normal operation, GoEnrichWS listens for requests from Sidekick. After parsing the input parameters the servlet calls the method for the Ontologizer classes with the

proper population, study set, and parameters. GoEnrichWS translates the returned terms back into MeSH terms and returns the results to Sidekick to be added to the AdvancedDataGrid.

3.5.3 Enrichment for gene expression

Gene expression refers to the number of transcripts produced from a gene, which is loosely related to the number proteins produced. Sets of genes that are over-expressed or under-expressed under a specific condition as compared to the population as a whole may be related. The European Bioinformatics Institute provides a web service, Gene Expression Atlas, within ArrayExpress [76] that contains curated data for gene expressions under different biological conditions across experiments. The conditions include cell type, developmental stage, and disease state among many others. *Enrich: for gene expression* allows input of a gene list and selection of multiple conditions. The URL call from Sidekick to ArrayExpress is managed by SideCache's WebServiceProxy and it takes the form:

`http://www.ebi.ac.uk/microarray-as/atlas.`

The web service call requires a species parameter and the condition searched for. There are 48 different conditions that were grouped into four different categories. A drop down list was created that allows selection of all elements in a category by checking the category checkbox, or when the category is opened, individual conditions within a category can be selected. It was found that ArrayExpress ignores any query with more than 100 genes. To handle this problem Sidekick calls ArrayExpress in groups of 50 genes. As ArrayExpress returns results they are parsed and added to the AdvancedDataGrid. After all results are processed the group counts are determined and displayed.

3.5.4 Enrichment for chromosomal proximity

Proximity of genes along the chromosome can indicate functional relationships between genes. For *Enrich: for chromosomal proximity* the user sets the *Max # of hits*, and Sidekick displays the gene groups that are most enriched for chromosomal proximity as determined by the number of base pairs separating the start positions of genes. Chromosome proximity enrichment could isolate targeted gene mutations as either a spontaneous or combinatorial event.

Chromosome proximity enrichment could also prove helpful in analyzing transcription factor targets.

Sidekick manages chromosomal proximity without any web service calls because the study set of genes has already been processed by either a previous query or by inputting a gene list. During that process, the gene information including chromosome and chromosome start position was downloaded. Sidekick iteratively combines genes into sets. Each iteration finds the nearest genes or gene sets and combines them into a single gene set. In this process eventually all genes on a single chromosome are grouped together. For each step, the newly formed group is placed into the AdvancedDataGrid.

3.5.5 Enrichment for gene-pairs

While gene list enrichment is very common in genomic studies, all of Sidekick gene enrichment analyses not only allow gene list enrichment, but also gene-pair lists. Sidekick pair-list enrichment analysis finds gene pairs where both elements are enriched for a specific term. As an example, *Enrich: for GO terms* determines if two genes localize to the same location in the cellular context. Predictions are greatly enhanced when corroborated with evidence connecting the protein pairs. If two genes that were found to interact are both enriched for chromosomal proximity and disease enrichment, this might be compelling evidence to target these genes for knockout when seeking drug related therapies.

Pair enrichment also provides evidence against gene interaction. In the case of GO component enrichment, if two proteins are not enriched for any components this means they do not localize to the same cellular component. This lack of locality provides evidence against interaction between the genes.

Sidekick manages pair enrichment by utilizing the appropriate enrichment strategy outlined previously however the study set of genes is a non redundant union of input and output genes from all pairs. To fill the AdvancedDataGrid, each pair from the input DataGrid is tested for membership in an enriched group. If both are present, this result is included in the AdvancedDataGrid. After all results are processed the group counts are determined and displayed.

3.6 Combining lists

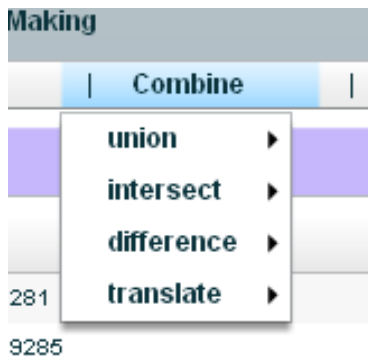


Figure 3.9 Combine options screenshot

Users can combine results from any of the queries that produce gene lists or gene-pair lists. Seen in Figure 3.9, the *union* and *intersect* operations combine any number of gene lists or gene-pair lists from the same species, while *difference* allows only two gene or gene-pair lists. The *union* finds all genes or gene pairs present in any input list, while *intersect* finds genes or gene pairs present in every input list. The *difference* operation asks the user to designate one of the input lists as the superset and removes all items of the other list from the superset.

The *translate* operation combines two pair lists when the output of the first list is comparable to the input of the second list. For example, suppose that the first list contains orthologs of human to mouse (pair A-B) and the second list contains interactions of genes in mouse (B-C). Translation produces a gene-pair list matching human genes to interacting genes in mouse (i.e. A→B and B→C gives A→C). For the current modules, column order represents input genes and discovered genes through a two-step process. By repeating the translation process with interacting gene-pair lists, one could find potential regulatory pathways. Eventually Sidekick will support using column order for analyses like transcription factor regulation where the first column holds the transcription factor and the second column holds the target of the transcription factor.

The flexibility that Sidekick offers users through the *Combine* operations enables even very complex research like InterologFinder. In Chapter 6 we show that with only a few mouse clicks, the work needed for orthologous transfer is trivial to accomplish with a combination of *translate*, *gene-gene Union gene-gene* and *gene-gene Intersection gene-gene* operations.

3.7 File management

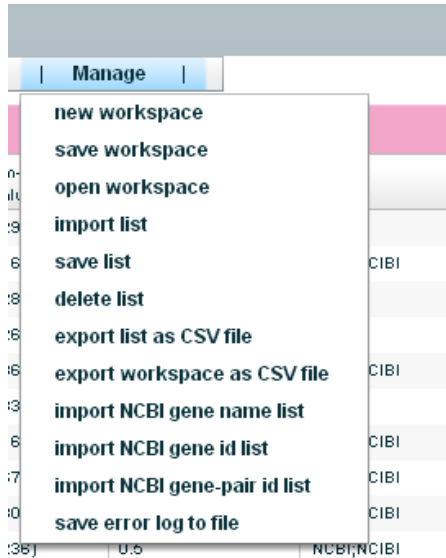


Figure 3.10 Manage options screenshot

File operations (see Figure 3.10) include saving and loading an entire workflow, saving and loading a gene or gene-pair list, importing NCBI IDs or NCBI gene names to form a user-created gene list, and exporting results as a comma separated file. The user may extract selected rows from a gene, gene-pair list or enrichment to create a subset list of the same type. In the case of enrichment, the duplicates add additional confidence to the combined confidence score as described in Chapter 5. For gene-pair lists, the subset can be either a row selection or an output column selection. In the case of row selection, the resulting subset is a gene-pair list, while column selection results in a single gene list comprised of all genes in the selected rows of the output column with any duplicates removed. Extracting rows from an enrichment results set produces the genes contained in the selected groups with duplicates removed. Sidekick uses XML as the underlying file format for most files.

3.8 Confidence and belief

Results			
Gene id (Name)	Score-from-source (Raw P-Value)	Score-from-source credibility	Sou
675(BRCA2)	25(0.00e-16)	0.5	NCB
8068(BRCATA)	1.301(5.00e-2)	0.5	NCB
2263(FGFR2)	7.44(3.63e-8)	0.5	NCB

Figure 3.11 Scores and credibility screenshot

Sidekick allows users to assert confidence or belief in results as they proceed through an analysis. Sidekick uses Dempster-Schafer Theory to combine scores and credibilities into a final confidence value (which users are free to ignore if they feel incredulous).

For example, NCIBI returns a p-value indicating the significance of its search results. Sidekick displays this p-value in the *Score-from-source* column of its *Results* section. The *Score-from-source credibility* indicates how reliable the user thinks this returned score is. Users can also assign credibility to the source itself as well as to individual results. Figure 3.11 shows an example of the *Score-from-source* and *Score-from-source credibility* columns.

User belief is fundamental to genomic analysis and will be covered in depth in Chapter 5.

CHAPTER FOUR: IMPLEMENTATION OF SIDEKICK

This chapter discusses Sidekick from the viewpoint of the implementation and shows how the Sidekick design enables it to be an extensible application.

4.1 Program Details

Sidekick is modular to enable future expansion and uses a single MXML file to handle its general graphical layout. The MXML file specifies the menu options and instantiates query objects. The Sidekick ActionScript classes can roughly be divided into three categories, query, cache and belief. The basic organization of the first two types of classes is summarized in Figure 4.1 and Table 4.1. Belief management is deferred until Chapter 5.

The queries all extend a base class which implements a unifying interface. They fill the Input panel with user input options required for the particular query. Upon activation (when the user presses Process Inputs), they initiate any web service calls and provide listeners for results and faults. When results are returned, the query parses the returned values, usually in XML, and stores the results in a grid object in the Results panel. The Results panel grids are richly functional and highly flexible. For standard data types, the columns are sortable, however user-defined sort functions can be applied to a column for user-defined data. For example the output for a gene includes the ID and gene name. When using the default sort, the order did not change due to the combination of two objects in one column. The sort function allowed one value to be sorted on. Further renderers can be added to columns of the Results grid to allow for unique display, for example the linking of gene ID/ name pairs to the NCBI website and the functionality of a radio button group used for belief management. Each query implements housekeeping operations like loading and saving its data to XML or a CSV file. Similar to the query modules, combination modules take their input from the current Results and combine them, replacing the current Results with combined information.

The second general type of class, the NcbiInfoCache, manages the local caching of information from NCBI. For example, when different queries return the same gene ID, Sidekick uses its cached value instead of using another web service to retrieve the information associated with the gene. A second manager of information, the ResultManager, maintains all of the Results including their icon representation and position in the workspace.

The final type of class manages belief and will be discussed in the next chapter. Figure 4.1 and Table 4.1 give overviews of the primary Sidekick classes and how each relates to the others.

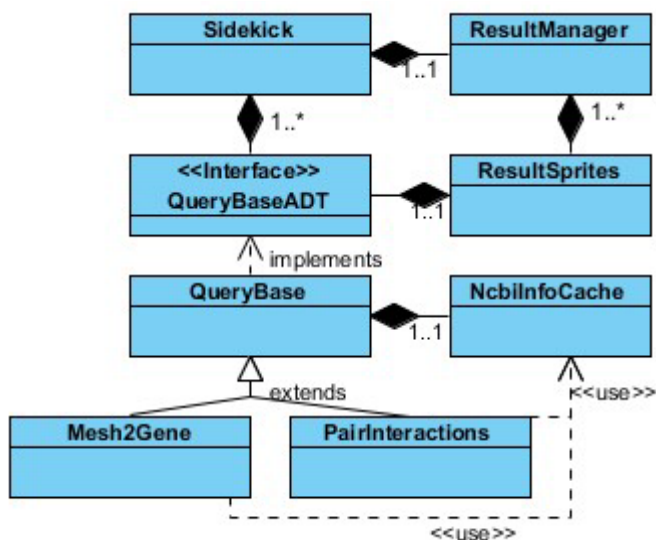


Figure 4.1 Top level Sidekick classes

Table 4.1 General description of Sidekick's top level classes

Class / Interface	Roles / Responsibility
<i>Sidekick</i>	<ul style="list-style-type: none"> • Manages the graphical content • Manages menu operations • Builds and adds <i>ResultSprites</i> to <i>ResultManager</i> • Instantiates child classes that extend <i>QueryBase</i>; • Calls child class methods in response to user actions
<i>ResultManager</i>	<ul style="list-style-type: none"> • Manages list of <i>ResultSprites</i>
<i>ResultSprites</i>	<ul style="list-style-type: none"> • Contains reference to corresponding query class representing operations • Contains graphical information about icons
<i>NcbiInfoCache</i>	<ul style="list-style-type: none"> • Manages a collection of gene entities. A gene entity holds genetic data defining a specific gene
<i>QueryBaseADT</i>	<ul style="list-style-type: none"> • Interface specifies methods that define a query
<i>QueryBase</i>	<ul style="list-style-type: none"> • <i>QueryBase</i> implements <i>QueryBaseADT</i> • Contains common methods and data elements for all queries • Builds and adds gene entities to a static single copy of <i>NcbiInfoCache</i>
<i>Mesh2Gene</i> and <i>PairInteractions</i>	<ul style="list-style-type: none"> • Examples of classes that extend <i>QueryBase</i> • Contain methods and data specific to the query • Search and update <i>NcbiInfoCache</i>

4.2 Sidekick use case

We describe Sidekick's architecture in terms of what happens during a series of research tasks. We will outline the process for finding a gene list given a search term in Table 4.2 and then using that list as input for finding an interacting gene-pair list in Table 4.3.

Table 4.2 Analysis steps for disease → gene list

User action	Description
Start Sidekick http://visual.cs.utsa.edu/sidekick	<ul style="list-style-type: none"> • Browser loads Sidekick's Shockwave Flash file after downloading it from the visual server, and starts its local Flash interpreter to run Sidekick • Sidekick lays out the application as seen in Figure 3.2 • Sidekick instantiates new <i>ResultManager</i>
Click <i>Query:</i> <i>disease/term</i> → <i>gene list</i>	<ul style="list-style-type: none"> • Sidekick stores last query (null) • Sidekick determines which menu option was selected and instantiates a new <i>Mesh2Gene</i> object. The constructor: <ul style="list-style-type: none"> ○ takes the last query as input ○ instantiates static (only) copy of <i>NcbiInfoCache</i> if it doesn't exist ○ instantiates <i>DataGrid</i> and corresponding data provider <i>ArrayCollection</i> • Sidekick sets newly instantiate <i>Mesh2Gene</i> object as current query • Sidekick calls the <i>Mesh2Gene.addMyInputs</i> method to add user input objects to Inputs • Sidekick sets Process Inputs button and associated listener "active"
Type search term, select species, and select NCBI search strategy Click Process Inputs button	<ul style="list-style-type: none"> • Sidekick's button listener responds to click and calls <i>Mesh2Gene.checkInputs</i> method • <i>checkInputs</i> returns if something is entered in search term input box • Sidekick executes <i>Mesh2Gene.handleThisProcess</i> method if <i>checkInputs</i> is true / generates error if false • <i>handleThisProcess</i> <ul style="list-style-type: none"> ○ adds columns to the DataGrid ○ adds DataGrid to Results ○ builds query for NCBI and NCIBI MeSH2Gene including result and error handlers ○ executes the web service calls • After <i>handleThisProcess</i> finishes (the web services run asynchronously in separate threads) Sidekick <ul style="list-style-type: none"> ○ sets Process Inputs button inactive ○ builds <i>ResultsSprite</i> for storage in <i>ResultManager</i> that contains reference to this query and icon information ○ displays icon in Workspace ○ builds listener for mouse click of icon

	<ul style="list-style-type: none"> ○ re-colors panels to match icon ● NCBI result handler <ul style="list-style-type: none"> ○ parses XML result from NCBI, this is only the gene Ids ○ searches its <i>ArrayCollection</i> for gene Ids; if not already in <i>ArrayCollection</i> <ul style="list-style-type: none"> ▪ queries <i>NcbiInfoCache</i> for this gene; if present add reference to <i>ArrayCollection</i> ▪ if not in cache; adds partial gene entity to cache; adds partial gene entity to <i>ArrayCollection</i>; builds query to Gene2InfoWS web service with IDs; sets up listeners and executes call ○ if already in <i>ArrayCollection</i> adds to source information column for existing entry ● NCIBI result handler <ul style="list-style-type: none"> ○ parses XML result from NCIBI, this is only the gene names ○ searches <i>ArrayCollection</i> for gene name; if not already in <i>ArrayCollection</i> <ul style="list-style-type: none"> ▪ queries <i>NcbiInfoCache</i> for this gene; if present add reference to <i>ArrayCollection</i> ▪ if not in cache; adds partial gene entity to cache; adds partial gene entity to <i>ArrayCollection</i>; builds query to Gene2InfoWS web service with names; sets up listeners and executes call ○ if already in <i>ArrayCollection</i> adds to source information column for existing entry ● Gene2Id (ids) result handler and Gene2Id (names) result handler <ul style="list-style-type: none"> ○ parses XML results from Gene2Id ○ updates proper gene entity with additional genetic information ○ when entity is updated, the corresponding <i>ArrayCollection</i> element is also updated because it is a reference to the entity
--	---

Table 4.3 Analysis steps for genes → interacting pair list

User action	Description
Selects <i>Mesh2Gene</i> results by clicking its icon in the Workspace	<ul style="list-style-type: none"> ● Sidekick <ul style="list-style-type: none"> ○ responds to click ○ queries ResultManager with icon; matching query is returned ○ sets current query to returned query ○ re-colors panels ○ executes current query's <i>restoreMyInputs</i> method; this method places user inputs into Inputs and restores the Results <i>DataGrid</i>

<p>Clicks Query genes → interacting pair list</p>	<ul style="list-style-type: none"> • Sidekick <ul style="list-style-type: none"> ○ stores the current query as last query (<i>Mesh2Gene</i>) ○ determines which menu option was selected and instantiates a new <i>PairInteractions</i> object. The constructor: <ul style="list-style-type: none"> ▪ takes the last query as input ▪ instantiates <i>DataGrid</i> and corresponding data provider <i>ArrayCollection</i> ○ sets newly instantiated object as current query ○ calls the <i>PairInteractions.addMyInputs</i> method which adds user input objects to Inputs ○ sets Process Inputs button and associated listener "active"
<p>Clicks Process Inputs button</p>	<ul style="list-style-type: none"> • Sidekick's button listener responds to click and calls <i>PairInteractions.checkInputs</i> method • <i>checkInputs</i> returns if something is entered in search term input box • Sidekick executes <i>PairInteractions.handleThisProcess</i> method if <i>checkInputs</i> is true / generates error if false • <i>handleThisProcess</i> <ul style="list-style-type: none"> ○ adds columns to the DataGrid ○ builds query for Gene2InfoWS and NCIBI MiMI including all the genes from the last query accessed by last query's <i>getGridProvider</i> method ○ builds result and error handlers ○ executes the web service calls • After <i>handleThisProcess</i> finishes (the web services run asynchronously in separate threads) Sidekick <ul style="list-style-type: none"> ○ sets click Process Inputs button inactive ○ builds <i>ResultsSprite</i> for storage in <i>ResultManager</i> that contains reference to this query and icon information ○ displays icon in Workspace ○ builds listener for mouse click of icon ○ re-colors panels to match icon • Gene2InfoWS result handler <ul style="list-style-type: none"> ○ parses XML result from Gene2InfoWS, which includes not only interactants but their gene information too ○ if new pair, adds results to ArrayCollection which updates DataGrid ○ if pair exists, update source information with NCBI ○ Adds output genes to <i>NcbiInfoCache</i> as appropriate • NCIBI MiMI result handler <ul style="list-style-type: none"> ○ Parses XML result from NCIBI, this is only the gene names ○ if new pair <ul style="list-style-type: none"> ▪ queries <i>NcbiInfoCache</i> for this output gene; if present add reference to ArrayCollection ▪ if not in cache; adds partial gene entity to cache; adds partial output gene entity to ArrayCollection; builds query to Gene2InfoWS

	<p>web service with names; sets up listeners and executes call</p> <ul style="list-style-type: none"> ○ if not new pair update source ● Gene2IdWS (names) result handler <ul style="list-style-type: none"> ○ Parses XML results from Gene2IdWS ○ updates proper gene entity with additional genetic information ○ when entity is updated, the corresponding ArrayCollection element is also updated because it is a reference to the entity
--	---

Many details of the implementation are left out of the previous use case, however the primary operations described are generally similar for all of the queries. Each is a derived class from the QueryBase which implements the QueryBaseADT interface. Each query manages its own processes and data unique to that query. Data universal to all queries like the DataGrid, ArrayCollection, icon color, and last query is stored as local objects in the QueryBase class. Methods universal to all queries like getGridProvider and getSpecies are implemented in the QueryBase class.

4.3 Sidekick – SideCache communication use case

We now describe the communication during the Mesh2Gene use case of Table 4.2 to clearly illustrate the steps used to handle WebServiceProxy and servlet communication between SideCache and Sidekick.

Table 4.4 Web service call steps and implementation description

Executing Method	Description
<i>handleThisProcess</i>	<ul style="list-style-type: none"> ● NCBI web service using WebServiceProxy in SideCache <ul style="list-style-type: none"> ○ build query http://eutils.ncbi.nlm.nih.gov/entrez/eutils/efetch.fcgi?db=gene&term="+MeSHTermNoSpaces+"AND"+getFullSpeciesName(<i>getSpecies()</i>)+"[Organism]" ○ encode this URL and prepend WebServiceProxy URL http://visual.cs.utsa.edu/WebServiceProxy/Proxy?url= ○ assign NCBI interaction result listener ○ assign generic restart fault listener (QueryBase) ○ register the NCBI interaction result listener with URL of query in <i>QueryBase's</i> resultHandler collection ○ send the web service ● NCIBI MeSH2Gene web service using WebServiceProxy <ul style="list-style-type: none"> ○ build query http://gene2MeSH.ncibi.org/fetch.php?MeSH="+MeSHTermNoSpaces+"&taxid="+<i>getSpecies()</i>

	<ul style="list-style-type: none"> ○ encode this URL and prepend WebServiceProxy URL http://visual.cs.utsa.edu/WebServiceProxy/Proxy?url= ○ assign Ncibi interaction result listener ○ assign generic restart fault listener (<i>QueryBase.restartfaultHandler</i>) ○ register the Ncibi interaction result listener with URL of query in <i>QueryBase's</i> resultHandler collection ○ send the web service
<i>WebServiceProxy.doGet</i>	<ul style="list-style-type: none"> ● WebServiceProxy <ul style="list-style-type: none"> ○ looks for this URL in its cache ○ tests for results in the cache: <ul style="list-style-type: none"> ▪ if present, return them to Sidekick's waiting result handler ▪ not present, add URL to cache and proceed ○ if not present tests if the maximum number of calls: <ul style="list-style-type: none"> ▪ not reached, execute call and add URL to cache ▪ reached, return fault to Sidekick ○ Note: each <i>doGet</i> request runs in a separate thread managed by Tomcat
<i>QueryBase.restartfaultHandler</i>	<ul style="list-style-type: none"> ● handles fault (either exceeded the number of calls to this URL by all users or the outside web service did not return a proper result) ● uses the fault's URL to find the matching result handler (In this example, this method exists in <i>Mesh2Gene</i>) and retries if retry counter not exceeded ● tests maximum number of retries <ul style="list-style-type: none"> ○ reached: query dies; the information is never added to the result grid and the error is written to the error log file ○ not reached: <ul style="list-style-type: none"> ▪ assigns the result listener and generic restart fault listener (this method) ▪ increments retries counter for this URL ▪ waits five seconds times number of retries ▪ executes web service call
<i>NcibiInteraction.resultHandler</i> and <i>NcibiInteraction.resultHandler</i>	<ul style="list-style-type: none"> ● Both methods <ul style="list-style-type: none"> ○ parse results and start a web service call to the Gene2InfoWS servlet managed by SideCache ○ build a query http://visual.cs.utsa.edu/Gene2InfoWS?operation=getInfo&sourceSpecies=9606&genes="+genes ○ assign Gene2InfoWS result listener ○ assign generic restart fault listener (<i>QueryBase.restartfaultHandler</i>) ○ register the Gene2InfoWS interaction result listener with URL of query in <i>QueryBase's</i> resultHandler collection ○ set the retries counter to 0 ○ send the web service

<i>Gene2InfoWS resultHandler</i>	<ul style="list-style-type: none"> • parses results and follows steps to place in DataGrid's ArrayCollection
----------------------------------	---

Although the two types of communication are quite different, the calls through WebServiceProxy and the individual web service servlets are handled in very similar ways. For both, using a generic restart fault handler allows for a unified approach for error handling. The result handlers of both types are constructed in the queries because of the differences in parsing to be done, and processing of information specific to the query.

In Chapter 3 the data behind the servlets managed by SideCache was described. Because several clients can access these servlets simultaneously and web service calls are not synchronized, care was taken to ensure proper function of the services. When possible local copies of Java's HashMap were used to store data and these were wrapped in Java's Collections.SynchronizedMap which guarantees synchronous use of the underlying HashMap data structure. When it was not feasible to store the data in memory, files were used. Because the files are read-only and the file pointers are declared in the local copy of the *doGet* method, access is synchronous and safe.

4.4 Sidekick extension use case

We now describe the steps needed for adding an additional query to illustrate the architecture and expandability of Sidekick. Gene regulation involves an interaction between a transcription factor protein, which we will identify by the gene that produces the protein, and the section of chromosome that contains the target gene. Often the easiest way to implement a new class is to use a similar one as a template. In this case, the *Query: gene → interacting pair list* query is a very good match.

Table 4.5 Steps to create a new query

Classes to modify	Description of programming done
main MXML file	<ul style="list-style-type: none"> • add the query's title • add to the menu listener, the <i>TF_TFT</i> query instantiation and standard code • add code to handle file input and output for the new class under file operation handling • Note: all code will be very much like the pair interaction query and most coding can be accomplished by copy and paste
<i>TF_TFT</i> query class	<ul style="list-style-type: none"> • insert a new ActionScript class that extends the <i>QueryBase</i>

	<p>class which implements the <i>QueryBaseADT</i> abstract data type. The <i>QueryBase</i> class provides much of the functionality required for any query generally falling into web service tasks (restarting on failure after wait, URL encoding), belief management (evidence updating, learning rate updating, credibility computation, button displays, clustering for visualization), and display (column comparison functions, grouping for enrichment modules, icon creation and storage).</p> <ul style="list-style-type: none"> • build constructor which takes the last selected query as input. • build <i>addMyInputs</i> method by filling the given input panel created within the MXML file with any user supplied information needed for the query, in this case only the user's notes text box is needed because the last result's gene list will be used for input. • build <i>restoreMyInputs</i> method by filling the input panel with the information from the <i>addMyInputs</i> method and the stored input results icon • build <i>addColumns</i> method by defining what will be displayed in the Results panel. Like the interaction query, both input and output genes will be included and rendered by the <i>UrlRenderer</i> class. This class allows linking to NCBI Entrez gene and displays the information as a hyperlink. This query will have the same columns as the interaction query so will be simple to copy and paste. • build File management methods, <i>toCSV</i>, <i>toXML</i>, <i>loadXML</i>, and <i>getSelectedRows</i>. These are very similar between like queries, but specific to the types of inputs required by the query. Again, a simple copy and paste from interacting pairs should suffice. • build <i>handleThisProcess</i> method which populates the grid, adds and registers appropriate belief management buttons and initiates all the web service calls. Because the grids are the same for the two queries, the interaction query code can be used for the first three. • build the handlers that process the results from the web service calls. The URL call, fault, and results handlers are similar to the pair interactions and have been detailed in the previous section. • If multiple public data sources provide transcription factor / transcription factor target information, a similar process of determining the correct query, parsing the results in a listener and adding to the data collection will take place. • The only conflict exists when the same tf/tf target is returned from both sources and in that case, rather than adding a redundant entry the evidence of the already existing result is updated with the additional source and the credibility adjusted.
--	---

<i>ResultManager</i> , <i>CombineResults</i> , <i>ProbabilityGraph</i> , <i>RadioButtonRenderer</i> , and <i>SelectionEvent</i>	<ul style="list-style-type: none">• For our example, none of these need to be modified.<ul style="list-style-type: none">○ The <i>ResultManager</i> will handle storage of the icon information and a reference to the query.○ <i>CombineResults</i> will handle the combining of confidence measurements to find a single final confidence value.○ <i>ProbabilityGraph</i> handles the Belief Manager appropriate for score-from-source and evidence.○ <i>RadioButtonRenderer</i> and <i>SelectionEvent</i> work together to handle click events on the radio buttons that enable iterative evaluation of results.
---	--

The wide variety of existing queries enables reuse of code and coding through example. Even in the primary Sidekick.mxml file, the code specific to each query is similar and when a similar query is used as a template, easy to mimic. Through the use of the base and abstract classes, new modules are already functionally rich and indicate what needs to be implemented for integration with Sidekick. Thus, Sidekick will be relatively easy to extend to add queries.

CHAPTER FIVE: BELIEF MANAGEMENT

A person's belief is influenced by their experiences, and attempts to use a one-size-fits-all generally alienate experts who disagree with the conclusions framed by the belief assignments made to describe findings. This does not mean that all belief measurements should be ignored, because often quantifying information allows for more informed decisions. Sidekick uses a flexible user-driven system of belief management.

5.1 Overview of belief and confidence difficulties

Sidekick manages a variety of confidence measurements to quantitatively describe belief in information and uses Dempster-Shafer's ability to represent uncertainty and combine knowledge. For example, an analysis might assign probabilities based on a decision tree trained on patient information features such as age, menopausal condition, breast quadrant and tumor size and use this information to predict whether a cancer will recur with a given probability ($P(\text{Recur})$). The standard probabilistic assumption is that the probability of remaining cancer-free is $1 - P(\text{Recur})$. However, this assumption removes uncertainty from the conclusion. In a clinical setting, a physician might deliver one of three predictions to a patient, “the tumor will likely recur”, “the tumor should not recur”, or “I don't know”. When faced with uncertainty, the physician often orders additional tests.

P-values are often used to describe enrichment tasks where probability of the null hypothesis is measured. Goodman[77] explains the inherent problem of using p-values as a means of measuring belief and uses a Bayesian approach to convert p-values into a probability measurement. However, conversions and combinations of p-values require previous knowledge of the distribution of the input p-values, which may or may not be available.

Dempster-Shafer theory (DST) in combination with Sidekick's visualization of *Score-from-source* addresses the problem of uncertainty, combination of beliefs, and conversion of p-values when the underlying distribution is unknown. Dempster-Shafer's theory provides the ability to model and combine certainty even in the presence of uncertainty. Multiple belief functions may be combined by applying DST iteratively for each basic confidence measurement assignment. When multiple web services return the same gene, DST increases confidence in that gene's inclusion in the results list.

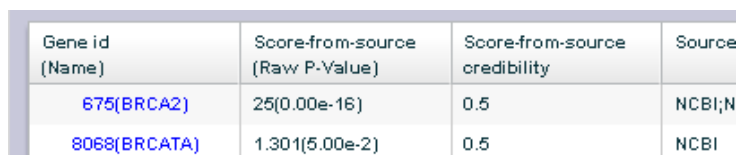
The Sidekick Belief Manager addresses the problem of p-value conversion without distribution information. For DST to be used, one must have beliefs assigned to each of the values represented in the data. Sidekick's visualization and clustering methodology gives the user the ability to manage the conversion from p-value or other metrics to credibility measures, reflecting prior knowledge in the credibility scores. Once credibility scores are assigned, either as a result of a computation or by the user, Sidekick uses DST to combine these scores into an overall result credibility.

5.2 Raw scores and credibility

Sidekick sources return different types of quality measures depending on the resource. For example the NCIBI MeSH2Gene web service, returns the number of articles supporting an association of gene to MeSH term as well as a p-value. Some sources, such as NCBI, do not return a quality measure at all. Different types of quality measures include: p-values, article counts, number of experiments supporting a result and group size.

A user's belief in the result produced by a search or analysis may be influenced by background knowledge and view of the data sources. Sidekick allows the user to adjust and manage several types of credibility. The level of belief or credibility for any single score is a value between -1 and 1, where -1 indicates complete skepticism and 1 indicates complete acceptance. Sidekick allows negative values to represent evidence against a particular result. Initially, Sidekick assigns a credibility of 0.5 to each score; which users are free to adjust as analysis proceeds.

5.2.1 Score-from-source and Score-from-source credibility



Gene id (Name)	Score-from-source (Raw P-Value)	Score-from-source credibility	Source
675(BRCA2)	25(0.00e-16)	0.5	NCBI;N
8068(BRCATA)	1.301(5.00e-2)	0.5	NCBI

Figure 5.1 Score-from-source screen shot

Each query has values used to influence the final credibility (*Combined credibility*) in the result. One basis for the credibility measurement is the score returned from the web service that provides the information. We outline the scores for each query and their format.

Each result entry from *Query: disease / term* → *gene list* includes a *Score-from-source* representing the significance of the association between the input disease or search term and

genes derived from PubMed abstracts. Gene2MeSH returns a p-value for this score based on the chance that the gene and associated search term were found together by randomly connecting genes and terms. NCBI's *ESummary* web service produces a gene list given a term (not necessarily a disease term), but does not assign a p-value to the gene-term pair. Sidekick uses a value of 0.05 as a default p-value for NCBI's *Score-from-source*. Sidekick uses NCBI's p-value when both sources return a particular gene-term pair.

Query genes → interacting pair list utilizes NCBI's MiMI [12]. MiMI was created by compiling several publicly available data sources including HPRD [66], IntAct [46], BIND [48], BioGRID [72], MINT [67], CCSB [73], DIP [65], Reactome [70], and MDC [74]. MiMI has also compiled the number of PubMed articles that refer to each interaction. The *Score-from-source* is the number of articles reported from MiMI.

Query genes → orthologous pair list maps genes from one species to corresponding genes in another species. Sidekick uses Ensembl's orthologous gene lists and displays the percent identity between the two orthologs as the *Score-from-source*.

For *Enrich for GO terms* the *Score-from-source* is the p-value representing the likelihood that the shared GO terms of the gene subset could have happened in a randomly chosen subset. This computation compares the study set and the population set in which all similar genes are included.

The outputs for *Enrich for disease terms* are similar to those of GO term enrichment, with the *Score-from-source* as the p-value representing the likelihood that a subset with shared MeSH terms happened randomly as compared to the general population.

Sidekick uses European Bioinformatics Institute (EBI)'s web service Gene Expression Atlas, within ArrayExpress for calculation of enrichment based on gene expression in the *Enrich for gene expression* module. The *Score-from-source* is the difference between the number of up regulated and down regulated experiments returned from ArrayExpress. By using the difference we find both the magnitude of the difference and because the difference is negative or positive we also can determine the type of regulation that is dominant.

Sidekick uses NCBI's gene chromosomal location for its *Enrich for chromosomal proximity* module. The gene groups that are most enriched for chromosomal proximity as determined by the number of base pairs separating the start positions of genes are retrieved. Using genome wide data from NCBI, Sidekick computes the *Score-from-source* as the likelihood

that the grouped set were found together in their range of nucleotides as set by the furthest starting positions compared to the entire chromosome length.

The *score-from-source* is for enrichment of gene-pair lists calculated in the same way as for a gene list. The score is recorded for the pair as indicated by the score assigned to the group that contains both genes in the pair.

The *Score-from-source credibility* indicates how much you believe the results based on the scores returned from the source. You can adjust these scores using the Belief Manager outlined in the next section. The inputs into the Belief Manager must be positive real numbers ordered from worst score (lowest value) to best score (highest value). For the *Scores-from-source* that are p-values, the raw score is converted by taking the negative log of the p-value. As you see in Figure 5.1 the converted score is displayed with the raw p-value in parentheses. The only other score that is adjusted is the *Score-from-source* for the *Enrich for gene expression module*. The raw score is the difference between the number of up regulated and down regulated experiments returned from ArrayExpress, however this allows for negative values. The direction of the regulation is ignored in the Belief Manager by taking the absolute value of the difference.

5.2.2 Source and source credibility

Another way to influence the overall result credibility is to evaluate the credibility of the sources that produce a gene or gene-pair list (*Source credibility*). Each source for a given query or filter is assigned equal weight of 0.5. The user can change this value to reflect belief in the data source in two distinct ways. By pressing the *Adjust source credibility* button, the user can globally change the confidence from 0.5 to reflect increased belief (> 0.5) or increased disbelief (< 0.5) in the reliability of the specific source. The other iterative method for specifying belief in the data source comes from belief or disbelief in individual results. When the user changes the radio buttons at the far right of a result row (see Figure 5.2), Sidekick modifies both the *Combined credibility* for that item and also the *Source credibility* of the site that produced the result.

Source credibility	Combined credibility(0-1)	Item credibility low — high
0.76	0.88	<input type="radio"/> <input checked="" type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/>
0.49	0.75	<input type="radio"/> <input type="radio"/> <input type="radio"/> <input checked="" type="radio"/> <input type="radio"/>
0.76	0.88	<input type="radio"/> <input type="radio"/> <input checked="" type="radio"/> <input type="radio"/> <input type="radio"/>

Figure 5.2 Individual result screen shot

During the analysis, Sidekick iteratively learns a user's beliefs based on the evidence provided by individual credibility decisions. Using a methodology similar to the online training for spam filtering of Goodman *et al.*[78], Sidekick adjusts the *Source credibility score* in a positive or negative direction when the user changes a radio button corresponding to an item from a particular source away from neutral, see Figure 5.2. The learning rates have default settings of $[-0.030, -0.015, 0.0, 0.015, 0.03]$, corresponding to adjustment of the five radio buttons to the left or right of neutral. The user can modify these rates by selecting *Adjust source credibility learning rates*.

5.2.3 Size-of-group and size-of-group credibility

During enrichment, a subset of related genes is found from a general population. Often a p-value is assigned to represent the probability that the subset was found at random. This means that a subset with 5 out of 10 in the general population might have the same p-value as 2 out of 10. The Belief Manager enables the user to change *Size-of-group credibility* based on the number of elements in the enriched set. This like *source credibility* influences the overall result credibility when the scores are combined.

5.3 Belief Manager

A user's belief in input sources depends on many factors including the user's background knowledge and view of the data sources. Sidekick provides an intuitive interface (see Figure 5.3) for organizing these beliefs. In Sidekick, the level of user belief or credibility for any single score is a value between -1 and 1 , where 1 indicates complete acceptance of a result. A credibility value of -1 indicates strong skepticism or alternatively complete belief in the result's negation. Sidekick allows negative values for individual scores so that users can specify that a particular result provides evidence against something. By default, Sidekick assigns a credibility of 0.5 to each score; however, the user is free to adjust this credibility using the Belief Manager.

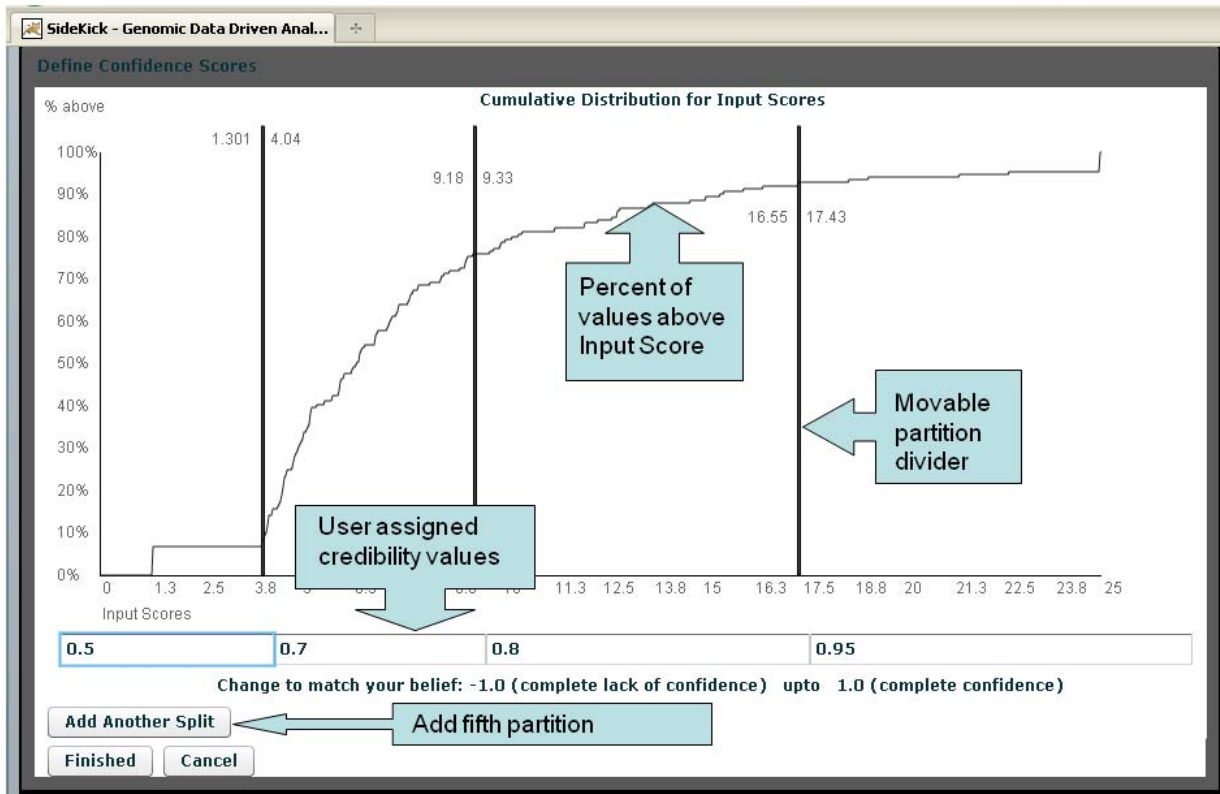


Figure 5.3 Screenshot of Belief Manager

Sidekick allows the user to control the level of belief in an interactive way. To visualize and assign user belief to a score the user presses the appropriate button. Sidekick displays the normalized scores using a cumulative probability density graph. The graph represents both the range of values and the relative density of specific values within the range. Dividers partition the input values and allow users to assign confidence scores to different groups of input values. By increasing or decreasing the number of partitions and moving the partitions, the user can change the credibility score assigned to the input values contained within a partitioned range. Sidekick uses the Expectation Maximization [79] algorithm for determining the initial *a priori* number of partitions and k-nearest neighbor [80] for clustering to a specified number of partitions (when adding or removing a single partition from the defined number of partitions).

5.4 Belief, disbelief and uncertainty

When the Belief Manager has been used, global source credibility adjusted and iterative evaluation of results using the radio buttons finished, each score has been converted into a

credibility measurement. The final step combines these score into a single Combined credibility score. When query results are combined, using the *Combine* operations the resulting scores are combined. However often one result does not have the same genes or gene-pairs as another. Handling missing data needs to be addressed.

Machine learning has been used to predict class membership including multiple and binary class problems. A simple binary class problem specifies belief in one result or the other, usually in the form of True or False conclusions. For example, we might consider the task of classifying a cancer as “Malignant” or “Benign”. Converting this problem into a True or False problem is simply rewording the problem as classifying a cancer as “Malignant” with true indicating it is malignant and false indicating it is not malignant. A diagnostic test can produce a conclusion about the cancer. However, it is important in medical diagnosis to increase confidence in conclusions through multiple tests. It is also important to recognize the differences in predictive quality among diagnostic tests.

In order to combine tests to form a single diagnostic conclusion two tasks must be accomplished. Belief and uncertainty must be modeled and combined. Dempster-Shafer’s Theory gives us this ability.

5.5 Dempster – Shafer Theory and *Combined credibility*

In a simple binary class problem the traditional probabilistic approach would have the probability of one class + probability of another = 1. However when uncertainty is also modeled this is not longer the case. The following discussion is taken from Adrian O'Neill's tutorial (<http://www.aonaware.com/dsetheory.htm>) on Dempster-Shafer Theory (DST).

DST defines the possible predictive outcomes using the term Universe of Discourse Θ also called the Frame of Discernment. Consider a case of two possible, mutually exclusive outcomes. Θ becomes $\Theta = \{T, F\}$ where T = true and F = false. DST also models exceptional situations as \emptyset . In this case, the Frame of Discernment is $\Theta = \{T, F, \emptyset\}$.

Elements of 2^Θ i.e. the set of all subsets of Θ are the class of general propositions in the domain. For our Frame of Discernment the possible sets are $\{T\}$, $\{F\}$, $\{TF\}$, and $\{\emptyset\}$. A function $m: 2^\Theta \rightarrow [0,1]$ is called a *basic probability assignment* if it satisfies:

$$m(\emptyset) = 0, \quad \sum_{A \subseteq \Theta} m(A) = 1 \quad \text{and } m(A) \geq 0 \text{ for all } A$$

in other words each possible prediction is assigned a probability, there are no exceptional situations, and collectively the probabilities add up to one. The quantity $m(A)$ is defined as A's *basic probability number*. It represents our exact belief in the proposition represented by A. The probability assigned to the set $\{TF\}$ indicates uncertainty. If there is no uncertainty *i.e.* $m(TF) = 0$ then $m(T) + m(F) = 1$, a situation that mimics a traditional machine learning probability assignment. Table 5.1 shows possible values of A with two different probability assignments m_1 and m_2 .

Table 5.1 Basic probability assignments, beliefs and subsets

A	m_1	m_2	$Bel_{m_1}(A)$	$Bel_{m_2}(A)$	subsets used for $m_1 \oplus m_2$
\emptyset	0	0	0	0	$\{\}$
$\{T\}$	0.5	0.75	0.5	0.75	$\{T_{m_1}\} \cap \{T_{m_2}\}, \{T_{m_1}\} \cap \{TF_{m_2}\}, \{TF_{m_1}\} \cap \{T_{m_2}\}$
$\{F\}$	0	0	0	0	$\{F_{m_1}\} \cap \{F_{m_2}\}, \{F_{m_1}\} \cap \{TF_{m_2}\}, \{TF_{m_1}\} \cap \{F_{m_2}\}$
$\{TF\}$	0.5	0.25	1.0	1.0	$\{TF_{m_1}\} \cap \{TF_{m_2}\}$

A belief function assigns to each subset of Θ a measure of our total belief in the proposition represented by the subset. There is a one-to-one relationship between belief functions and basic probability assignments. They are related by the following two formulae:

$$Bel(A) = \sum_{B \subseteq A} m(B)$$

you can recover the probability assignment:

$$m(A) = \sum_{B \subseteq A} (-1)^{|A-B|} Bel(B)$$

Thus, a belief function and a basic probability assignment convey exactly the same information. The belief $Bel(A)$ for a set A is defined as the sum of all the basic probability of subsets of the set of interest. Notice from the Frame of Discernment $\Theta = \{T, F, \emptyset\}$ the *basic probability assignment* of the possible subsets $m\{T\}$ and $m\{T,F\}$ influence $Bel(T)$.

Two pieces of evidence can be combined using DST. Let m_1 and m_2 be basic probability assignments on the Frame of Discernment Θ . Belief is combined by finding the orthogonal sum of m_1 and m_2 , $m = m_1 \oplus m_2$ and $m(\emptyset) = 0$

$$m(A) = K \sum_{X \cap Y = A} m_1(X) \bullet m_2(Y) .$$

$$K^{-1} = \sum_{X \cap Y \neq \emptyset} m_1(X) \bullet m_2(Y)$$

The quantity $\log K$ quantifies the conflict between the two basic probability assignments.

Sidekick includes confidence measures for all of the filters, queries, and combinations and combines multiple evidence using Dempster – Schafer theory to produce a final *Combined credibility* for each result in a list and for combined results.

It is helpful to look at an example, given our previous example where enrichment produces two different subsets of enrichment one a subset with 5 out of 10 in the general population with the same p-value as 2 out of 10. If the p-value was small compared to the other enriched sets the user might assign a belief score of 0.5 to the *Score-from-source credibility* of both sets. However, the same user might believe the set that contains half of the genes from the general population is more believable than the set with only one fifth of the genes. Using the Belief Manager, the user might assign the groups with size of 2 a belief of 0.5 and those of size 5 a belief of 0.75. This translates into Table 5.2 and 5.3. To draw connections between Table 5.1 and Table 5.2 let $m_1 \Leftrightarrow$ *Group size credibility*, $m_2 \Leftrightarrow$ *Score-from-Source Credibility*, $T \Leftrightarrow$ Belief, $F \Leftrightarrow$ Disbelief and $TF \Leftrightarrow$ Uncertainty.

Table 5.2 DST combination of stronger credibilities

			Group size credibility		
			Belief	Disbelief	Uncertainty
			0.75	0	0.25
Score from Source Credibility	Belief	0.5	0.375	0	0.125
	Disbelief	0	0	0	0
	Uncertainty	0.5	0.375	0	0.125
K = 1					
$m_{score-from-sourceCred} \oplus m_{groupSixCredibility} \{Belief\} = (1)(0.375+0.125+0.375) = 0.875$					
$m_{score-from-sourceCred} \oplus m_{groupSixCredibility} \{Disbelief\} = (1)(0+0+0) = 0$					
$m_{score-from-sourceCred} \oplus m_{groupSixCredibility} \{Uncertainty\} = (1)(0.125) = 0.125$					

Table 5.3 DST combination of weaker credibilities

			Group size credibility		
			Belief	Disbelief	Uncertainty
			0.5	0	0.5
Score from Source Credibility	Belief	0.5	0.25	0	0.25
	Disbelief	0	0	0	0
	Uncertainty	0.5	0.25	0	0.25
K = 1					
$m_{score-from-sourceCred} \oplus m_{groupSixCredibility} \{Belief\} = (1)(0.25+0.25+0.25) = 0.75$					
$m_{score-from-sourceCred} \oplus m_{groupSixCredibility} \{Disbelief\} = (1)(0+0+0) = 0$					
$m_{score-from-sourceCred} \oplus m_{groupSixCredibility} \{Uncertainty\} = (1)(0.25) = 0.25$					

Sidekick also uses Dempster-Shafer to combine credibility scores to obtain an overall belief credibility score from combined results. While the user is able to indicate disbelief with a negative credibility score, the *Combined credibility* indicates belief and therefore always ranges from 0 to 1 where 1 indicates perfect combined belief and 0 indicates lack of belief. These credibility scores allow users to focus on results that are likely to be more significant or more reliable. The previous example then becomes an illustration of combining results. If one disease → gene list included gene A in the results and after the user adjusted the *Score-from-source credibility* and the *Source credibility*, gene A was assigned a *Combined credibility* of 0.5. If the same gene was found in a different disease → gene list and its *Combined credibility* was 0.75, the resulting *Combined credibility* for gene A would be 0.88. If gene B was present in both results with *Combined credibility* of 0.5 for each result, its *Combined credibility* would be 0.75.

It is worth noticing, if a third gene C were only present in one result list, its final *Combined credibility* would be the same as the *Combined credibility* of the only result. Lack of evidence is not negative evidence in DST.

The user is also free to ignore completely the tracking of belief.

CHAPTER SIX: CASE STUDY

6.1 Introduction

This chapter highlights the strengths of Sidekick by reproducing the analysis of InterologFinder [1], which involves creation of a protein interaction network comprised of known and predicted interactions. It also demonstrates how to integrate belief in predicted interactions using gene enrichment information.

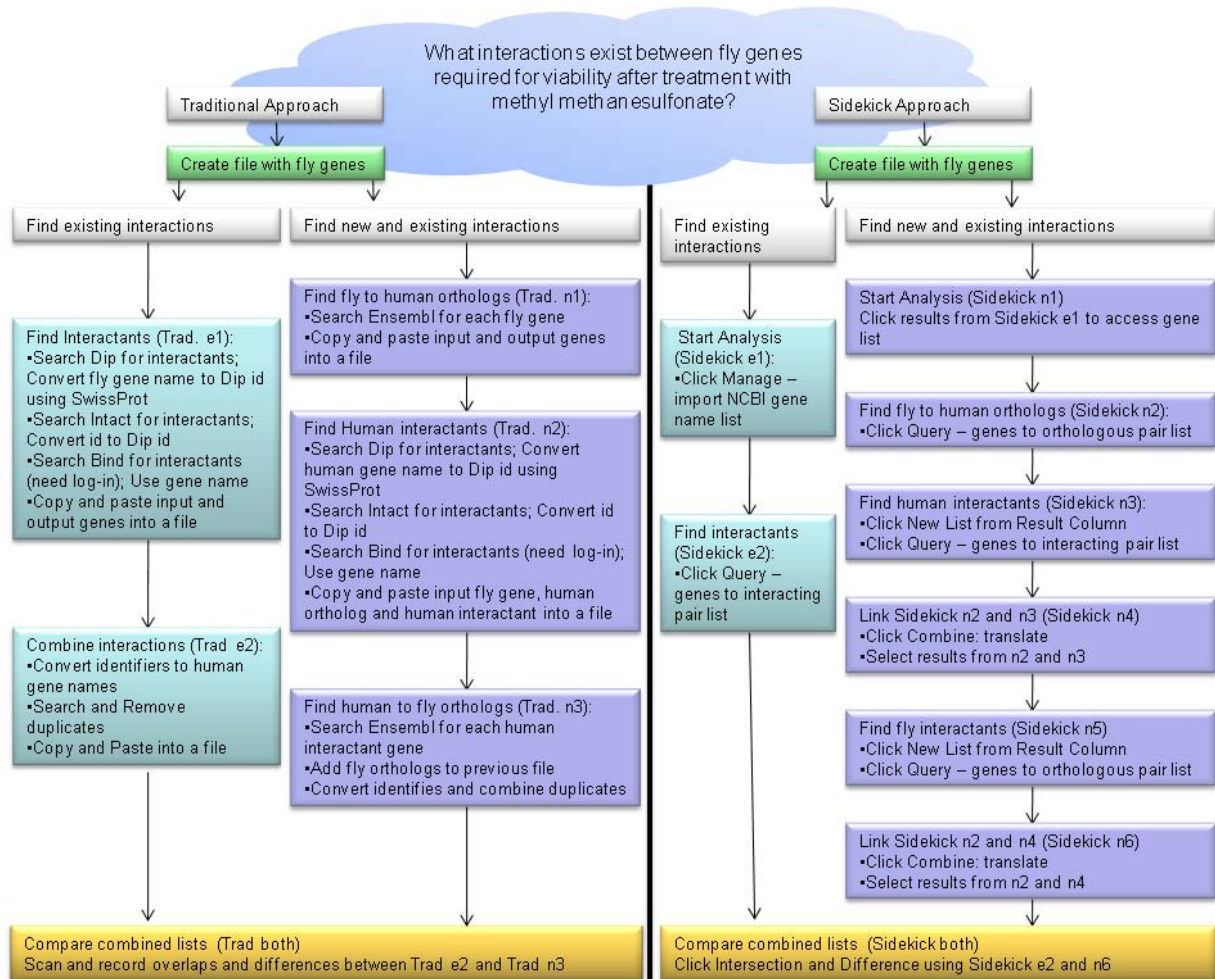


Figure 6.1 Comparison of research goals reached by hand and by using Sidekick

6.2 Sidekick compared with InterologFinder

Described in Figure 6.1, InterologFinder constructed a protein-protein interaction network comprised of known and predicted interactions within five species. InterologFinder inferred interactions in a target species by looking for known interactions in source species and then matching orthologous proteins in the target species. Team biologists then applied

coimmunoprecipitation to verify predicted interactions in fly. The biologists' understanding of protein conservation across speciation events enabled confidence in this approach.

InterologFinder also included a quantitative way to compare the interactions. Protein interactions range from weak to strong interactions. InterologFinder combined several measurements including the experimental support for a known interaction, the species support for known and predicted interactions, and the orthologous support of known and predicted interactions into a single score to represent the believability of both known and predicted interactions.

InterologFinder used a different method of evidence combination than Sidekick does and did not incorporate user belief.

As with Sidekick, InterologFinder analysis does not focus on maximizing the number of possible interactions by increasing coverage, but rather on generating as high quality interactions as possible. The work was carried out by a team comprised of both computer scientists and biologists, each bringing different perspectives to the analysis task. The computational tasks involved combining a variety of datasets, matching different identifiers for the same proteins, and predicting the protein interactions in one species by analyzing protein interactions among orthologous proteins in other species. The biologists brought specific expert knowledge to the process by determining which data sets should be trusted as high quality and the biological foundation for predicting interactions. They had formulated opinions through an understanding of the process by which the data repository incorporates new information and through a working knowledge of the data used for their own research and its worth as related to their work.

Sidekick enables computational analysis of systems such as InterologFinder and places researchers in control of their prior biological knowledge and research goals without the need for computational support. InterologFinder discovered a number of interactions that had not appeared in other data sources (specifically, MED26 – MED4, MED26 – MED16, and MED26 – MED17), which were subsequently verified in the wet lab. Users can easily reproduce the published results of InterologFinder using Sidekick's simple queries and data combination modules in 10 easy steps. We compare the tasks required for both development and use of InterologFinder and Sidekick. The discovery of multi-species protein networks requires orthologous gene lists and protein-protein interactions for the species to be studied (e.g., for the example, human, mouse, worm, fly and yeast).

6.3 Typical manual approach (as illustrated by InterologFinder)

Team biologists began the InterologFinder work with a small list of potential target genes of interest and determined interactions present in publicly available data bases (Figure 6.1, Trad e1 – e2). The computational team members expanded the interaction network by adding interactions present in orthologs based on orthologous relationships between genes specified by Ensembl (Figure 6.1, Trad n1 – n3). These relationships are identified by Ensembl identifiers that must be mapped to NCBI gene IDs. A computational scientist downloaded, parsed, and remapped these interactions in order to implement InterologFinder. This process was performed multiple times when Ensembl was updated (Figure 6.1, Trad. n1 and n3).

InterologFinder also combined several data sources (Bind, Dip and IntAct). Computational team members downloaded Ensembl synonym tables and converted protein identifiers from the IntAct and DIP databases into Ensembl IDs. The BIND database uses NCBI IDs. The data required remapping to the latest IDs and finding associated protein accession numbers, removing non-protein molecules, filtering for the appropriate species, and removing redundant results. Over the course of developing the application that predicted protein interactions, scientists downloaded, parsed and combined the data sources multiple times because the information became out-of-date (Figure 6.1, Trad. e1 and n2).

InterologFinder iteratively processes the interaction files of all five species using the following strategy. For a test gene-pair, G1 and G2, that does not have a known interaction in the target species, find orthologs (G1' and G2') in another species and check for interactions between the orthologs. Then use the orthologous relationship to infer the interaction for G1 and G2 (Figure 6.1, Trad. n1 – n3). Development of the program took many hours of programming and several hours to run. InterologFinder used this method to identify the predictions MED26 – MED4, MED26 – MED16, and MED26 – MED17 in fly from interactions in human. Scientists selected these gene interactions because of their inclusion in a genome-wide RNAi screen of genes required for viability after treatment with methyl methanesulfonate.

Comparing the lists in InterologFinder (Figure 6.1, Trad both) required a program to compare all known (Figure 6.1, Trad e2) and discovered interactions (Figure 6.1, Trad n3). When done by hand it is only feasible to check a few interactions for membership in both lists.

6.4 Sidekick's flexible exploratory approach

Sidekick provides a simple *Query: genes → orthologous pair list* that allows users to find the orthologs in a specified species. Behind the scenes, Sidekick uses the same files from Ensembl. However, these files are cached on the Sidekick server. Sidekick only downloads the information when a user requests a new species – species combination and refreshes the local copy of the two files weekly to maintain accuracy. Downloading of files is necessary because Ensembl does not offer web services for data retrieval; however Sidekick manages the data download (Figure 6.1, Sidekick n2 and n5).

Sidekick's *Query: genes → interacting pair list* uses files downloaded from NCBI, managed by the Gene2InfoWS of the SideCache communication suite, and NCBI's MiMI's web service to retrieve interaction information. MiMI contains all of the data sources included in InterologFinder's analysis and several additional interaction data sources. Sidekick doesn't download files, but rather acquires just-in-time information using MiMI web services.

Sidekick's directed research approach enables the user to achieve the same results as a process such as InterologFinder. For example, using the fly gene MED26 as input into the *Query: genes → orthologous pair list* with *H. sapiens* as the target species, the user will find the fly pair ortholog: 43816(MED26) – 9441(MED26) (Figure 6.1, Sidekick n2). The user can reduce the result to a single gene list by selecting *New List From Result Column*. The single human gene list now forms the input into *Query: genes → interacting pair list*, producing a second gene-pair list (Figure 6.1, Sidekick n3). This pair list contains 45 human interactions. The *Combine: translate: geneA-geneB translate geneB-geneC → geneA-geneC* connects the input column of the orthologs (Figure 6.1, Sidekick n2) to the output column of the interactants (Figure 6.1, Sidekick n3). The resulting pair list (Figure 6.1, Sidekick n4) has 45 fly genes and their orthologous interacting partners in human. After extracting the results column, using *Query: genes → orthologous pair list* (Figure 6.1, Sidekick n5), and translating between interactants and orthologs, we now have connected the original fly gene with fly interactants (Figure 6.1, Sidekick n6). This process finds both known and unknown interactions. To isolate only predicted interactions, simply use the original fly input list and run *Query: genes → interacting pair list* to form a pair list (Figure 6.1, Sidekick e2). The *Combine: gene-gene diff gene-gene* operation lists interactions found through orthologous transfer and known interactions removing all known interactions leaving only interactions not known in public data bases (Figure

6.1, Sidekick both). This list not only contains all three of the interactions validated by InterologFinder, but also 41 other interactions that might warrant further analysis. Figure 3.1 shows a screenshot of Sidekick at the end of a similar analysis.

6.5 Example of Sidekick's handling of belief

Forty-four possible gene interactions involving MED26 were determined using orthologous transfer. Using enrichment and user belief we further evaluate these possible interactions. Starting with *Enrich for GO terms* and selecting Component as the Go Type we find forty-two gene interactions enriched for some grouping by Component. There were 20 such groups. Using *Enrich for GO Term (Process)*, *Enrich for GO Term (Function)*, *Enrich for MeSH Term*, *Enrich for Chromosome Proximity*, and *Enrich for Gene Expression (Disease Progression)* the number of interactions found to be enriched varied from none of them to forty-one. Using *Combine: gene-gene diff gene-gene* with the complete list of predicted interactions as the superset and the interactions enriched for GO component as the subset, we are able to isolate the two interactions where the two genes never localized to the same cellular component. By assigning this result set a user confidence of -0.95 we are asserting our belief that for two genes to interact they must localize to the same cellular component. Keeping all other enrichments and the original list of predicted interactions with the default confidence scores of 0.5 we combine all of the evidence using *Combine: gene-gene union gene-gene*. The original list ensures all interactions will appear in the result and each enrichment provides evidence, negative in the special case. The resulting scores show the two interactions never found to localize together with very low confidence (0.04 and 0.05). The other interactions (including the ones verified by InterologFinder) all have high confidence scores of 1.0.

6.6 Discussion

Sidekick was developed to address issues with InterologFinder and other genomic analysis tools. We discuss a few that directly relate to InterologFinder and Sidekick.

6.6.1 Data downloads and processing

Even now after the publication of InterologFinder, to keep the website pertinent, the data on which the interaction network is created must be downloaded and reprocessed. This includes the data from Dip, Bind, IntAct and Ensembl. Unless an automatic system is in place, data will

not stay current. Sidekick's use of web services and automatic downloads of data to the server ensures that the information will continue to remain up-to-date and usable. Occasionally web services will change the calling URL or format of the results which will require modifications to Sidekick's modules; however this happens much less frequently than new data is available, for instance, NCBI, where data is updated daily.

6.6.2 Inflexible scoring

The complex scoring mechanism of InterologFinder combines homology, species count, and number of experiments into a single combined score. Through the use of multiple queries, source and score-from-source the same information can be quantified in Sidekick. Within a single species pairing, such as using human to predict mouse interactions, the number of sources that support the predicted interactions and the identity score for the orthologous relationships taken together with the user's belief form credibility measurements for that species transfer. Multiple queries involving different species would produce different lists of interactions found through homologous transfer. Combining these lists would take the individual species credibility measurements and form a multi-species credibility measurement much like the InteroScore found in InterologFinder. However, in Sidekick the user's belief is taken into account allowing increase confidence in one type of measurement over another.

Sidekick's flexible genomic discovery system enables even more sophisticated scoring of homologous transfer. As several interaction databases have done, in the analysis of section 6.5 we explored how enrichment could provide positive and negative evidence of the interactions. We saw that the interactions found and validated through InterologFinder were in fact given higher confidence because of their similar enrichment patterns including localization, chromosomal proximity, etc. Also interesting was the appearance of interactions found through orthologous transfer that had no similar enrichment pattern especially in the case of cellular localization. This lack of localization to the same cellular component should cause serious doubt in the prediction of an interaction.

6.6.3 Simple use of NCBI

In InterologFinder we recognized NCBI as a standard for genomic research and used it to prune data from Dip, Bind, and IntAct. We did this by taking the identifiers from the interaction databases, converting them into NCBI identifiers and, using a Perl script, tested their status in

NCBI as valid, discontinued or replaced. The script would run overnight and was re-run each time new data was downloaded to ensure the use of only valid identifiers. Sidekick's automatic download of NCBI gene data only finds identifiers that are current and not discontinued or replaced. The resulting entries for any Sidekick query are only current identifiers, ensuring quality data without the complex processing required in InterologFinder. Another simple, but important aspect of the use of NCBI identifiers is the ability to link to the NCBI website. While it is trivial to copy an identifier from a list of results and paste it into the appropriate area of NCBI Gene Entrez web site, it is much more convenient and faster to click the hyperlink within a gene or gene-pair result and have the appropriate NCBI web page load.

CHAPTER SEVEN: CONCLUSION AND FUTURE WORK

7.1 Conclusion

Sidekick offers an easy-to-use web-based, flexible alternative to current research tools. Belief updating through visualization allows scientists to incorporate their own background understanding into the manipulation of data. Gene and gene-pair list discovery, enrichment and combination allow accessibility to data and complex biological discovery. Sidekick is unique in its capabilities for scoring and manipulating gene-pair lists as well as its user-belief management. While the examples emphasized its role in initial discovery and exploration, Sidekick can also play an important role in interpretation of results obtained from other types of analyses. Often the final stage of an analysis such as clustering produces a gene list. In the discussion of the results, researchers typically apply enrichment analysis to explain how the resultant genes are related. Researchers can further explore these relationships by importing their gene lists into Sidekick and quickly determine interactions, enrichments, and orthology.

7.2 Extending Sidekick

We will continue to work with scientists using Sidekick to implement additional modules and services. Not only will these expand analysis involving genes, but also those involving other molecules such as proteins, and protein complexes. Specifically there is great interest in managing transcription factor (TF) and TF targets and microRNAs. TF genes mediate the creation of proteins from other genes by enhancing or inhibiting translation of the target. Transcription factors even regulate other transcription factors leading to a network of effects. Sidekick can easily manage these types of relationships using gene-pair results. Although the problem is complex because proteins and not the genes that produce the proteins do the vast majority of the cellular work, we look forward to including protein-specific queries. This inclusion would also give rise to additional enrichments like enrich for interacting domains. Another exciting development is the availability of a compound database that has compiled the chemical compounds known to affect change to specific genes/proteins. Adding this as an enrichment will enable scientists to explore possible drug interactions and other side effects.

Additional queries can also take the form of predictive algorithms written as web services on the server. These coupled with annotations from public data bases will expand coverage for

some problems. For example, when Sidekick expands to include proteins, ModECOC will be included as a means to annotate each protein with GO component information. Not only will this offer a broader view of an analysis, this also turns Sidekick into a platform to provide information to the public. When used by itself this query provides the user with information only known through our predictive algorithm and perhaps interesting for other work such as for transcription factor filtering.

Building a new query is relatively simple by using a similar query as a template. However the URL needed for the particular web service will be different and the returning document is generally different. We plan to allow users to specify these differences by a simple XML file that iteratively lists each web service, the structure of the returning data and what kind of result grid will be formed (currently single or gene-pair). By parsing these XML files at runtime and building the corresponding query, on the fly, Sidekick will become easily modifiable by anyone who knows the syntax of the XML file.

Extension of the server side caching system (SideCache) can take many forms. We look forward to improving query management by breaking some queries into individual parts. For example a typical NCBI query will be formed with many IDs in a single query (<http://visual.cs.utsa.edu/WebServiceProxy/proxy/?http://> etc genes=8906,8907,1425 etc. The existing caching system sees the URL as a single entry in the cache and would not equate it to a query with the same genes in a different order. By storing each Id individually and parsing future queries, the cache can return gene results individually in spite of the collective web service call.

Sidekick has already enabled research that was not expected. The discovery process that it enables is exciting and will continue to form a solid foundation for continued research and development as genomic information expands.

APPENDIX

Table APPENDIX.1 Query classes required by QueryBaseADT interface

```
* function addMyInputs(inputVB:VBox):void;
* function handleThisProcess(myPanel:VBox):void;
* function checkInputs():Boolean;
* function restoreMyInputs(inputVB:VBox,myPanel:VBox):void;
* function loadXml(inputs:XML):void;
* function toCSV():String;
* function toXML():String;
* function getSelectedRows():XML;
* function getSpriteType():int;
function getInputQuery():QueryBaseADT;
function getInputSpecies():String;
function setInputSpecies(ins:String):void;
function getIsEnrichment():Boolean;
function getIds():ArrayCollection;
function getLogFile():String;
function getSpecies():String;
function getFullSpeciesName(id:String): String;
function getGridProvider():ArrayCollection;
function getLabel():String;
function getInputResultSprite():ResultSprites;
function getColor():Number;
function setColor(n:Number):void;
function setInputResult (inResult:QueryBaseADT):void;
function setSpecies (s:String):void;
function setLabel(s:String):void;
function clearIds():void;
function hasResults():Boolean;
function doSaveResult():void;
function roundNumber (n:Number, dec:Number):Number;
function processFromGeneIds (idsString:String,p:VBox) : void;
function processFromGeneNames (idsString:String,p:VBox) : void;
function processFromGenePairsIds (idsString:String,p:VBox) : void;
```

* indicates overridden in derived class

BIBLIOGRAPHY

1. Doderer M, Wiles AM, Ruan J, Gu TT, Ravi D, Blackman B, Bishop AJ: **Building and analyzing protein interactome networks by cross-species comparisons.** *BMC Syst Biol* 2010, **4**:36.
2. Bruggeman FJ, Westerhoff HV: **The nature of systems biology.** *Trends Microbiol* 2007, **15**:45-50.
3. Giglia E: **New year, new PubMed.** *Eur J Phys Rehabil Med* 2009, **45**:155-159.
4. Flicek P, Aken BL, Ballester B, Beal K, Bragin E, Brent S, Chen Y, Clapham P, Coates G, Fairley S, et al: **Ensembl's 10th year.** *Nucleic Acids Res* 2010, **38**:D557-562.
5. Matys V, Fricke E, Geffers R, Gossling E, Haubrock M, Hehl R, Hornischer K, Karas D, Kel AE, Kel-Margoulis OV, et al: **TRANSFAC: transcriptional regulation, from patterns to profiles.** *Nucleic Acids Res* 2003, **31**:374-378.
6. Castellano M, Mastronardi G, Bellotti R, Tarricone G: **A bioinformatics knowledge discovery in text application for grid computing.** *BMC Bioinformatics* 2009, **10 Suppl 6**:S23.
7. Pounds S, Cheng C, Cao X, Crews KR, Plunkett W, Gandhi V, Rubnitz J, Ribeiro RC, Downing JR, Lamba J: **PROMISE: a tool to identify genomic features with a specific biologically interesting pattern of associations with multiple endpoint variables.** *Bioinformatics* 2009, **25**:2013-2019.
8. Du Z, Li L, Chen CF, Yu PS, Wang JZ: **G-SESAME: web tools for GO-term-based gene similarity analysis and knowledge discovery.** *Nucleic Acids Res* 2009, **37**:W345-349.
9. Ashburner M, Ball CA, Blake JA, Botstein D, Butler H, Cherry JM, Davis AP, Dolinski K, Dwight SS, Eppig JT, et al: **Gene ontology: tool for the unification of biology. The Gene Ontology Consortium.** *Nat Genet* 2000, **25**:25-29.
10. Bindea G, Mlecnik B, Hackl H, Charoentong P, Tosolini M, Kirilovsky A, Fridman WH, Pages F, Trajanoski Z, Galon J: **ClueGO: a Cytoscape plug-in to decipher functionally grouped gene ontology and pathway annotation networks.** *Bioinformatics* 2009, **25**:1091-1093.
11. Ramos H, Shannon P, Aebersold R: **The protein information and property explorer: an easy-to-use, rich-client web application for the management and functional analysis of proteomic data.** *Bioinformatics* 2008, **24**:2110-2111.
12. Jayapandian M, Chapman A, Tarcea VG, Yu C, Elkiss A, Ianni A, Liu B, Nandi A, Santos C, Andrews P, et al: **Michigan Molecular Interactions (MiMI): putting the jigsaw puzzle together.** *Nucleic Acids Res* 2007, **35**:D566-571.
13. Kanehisa M, Goto S, Furumichi M, Tanabe M, Hirakawa M: **KEGG for representation and analysis of molecular networks involving diseases and drugs.** *Nucleic Acids Res* 2010, **38**:D355-360.
14. Cochrane GR, Galperin MY: **The 2010 Nucleic Acids Research Database Issue and online Database Collection: a community of data resources.** *Nucleic Acids Res* 2010, **38**:D1-4.
15. Jensen LJ, Kuhn M, Stark M, Chaffron S, Creevey C, Muller J, Doerks T, Julien P, Roth A, Simonovic M, et al: **STRING 8--a global view on proteins and their functional interactions in 630 organisms.** *Nucleic Acids Res* 2009, **37**:D412-416.

16. Huang da W, Sherman BT, Lempicki RA: **Systematic and integrative analysis of large gene lists using DAVID bioinformatics resources.** *Nat Protoc* 2009, **4**:44-57.
17. Dennis G, Jr., Sherman BT, Hosack DA, Yang J, Gao W, Lane HC, Lempicki RA: **DAVID: Database for Annotation, Visualization, and Integrated Discovery.** *Genome Biol* 2003, **4**:P3.
18. Rowe A, Kalaitzopoulos D, Osmond M, Ghanem M, Guo Y: **The discovery net system for high throughput bioinformatics.** *Bioinformatics* 2003, **19 Suppl 1**:i225-231.
19. Shannon PT, Reiss DJ, Bonneau R, Baliga NS: **The Gaggle: an open-source software system for integrating bioinformatics software and data sources.** *BMC Bioinformatics* 2006, **7**:176.
20. Bare JC, Shannon PT, Schmid AK, Baliga NS: **The Firegoose: two-way integration of diverse data from different bioinformatics web resources with desktop applications.** *BMC Bioinformatics* 2007, **8**:456.
21. Reich M, Liefeld T, Gould J, Lerner J, Tamayo P, Mesirov JP: **GenePattern 2.0.** *Nat Genet* 2006, **38**:500-501.
22. Matos S, Arrais JP, Maia-Rodrigues J, Oliveira JL: **Concept-based query expansion for retrieving gene related publications from MEDLINE.** *BMC Bioinformatics* 2010, **11**:212.
23. Killcoyne S, Carter GW, Smith J, Boyle J: **Cytoscape: a community-based framework for network modeling.** *Methods Mol Biol* 2009, **563**:219-239.
24. Shannon P, Markiel A, Ozier O, Baliga NS, Wang JT, Ramage D, Amin N, Schwikowski B, Ideker T: **Cytoscape: a software environment for integrated models of biomolecular interaction networks.** *Genome Res* 2003, **13**:2498-2504.
25. Morcos F, Lamanna C, Sikora M, Izaguirre J: **Cytoprophet: a Cytoscape plug-in for protein and domain interaction networks inference.** *Bioinformatics* 2008, **24**:2265-2266.
26. Xia K, Dong D, Han JD: **IntNetDB v1.0: an integrated protein-protein interaction network database generated by a probabilistic model.** *BMC Bioinformatics* 2006, **7**:508.
27. Mathivanan S, Periaswamy B, Gandhi TK, Kandasamy K, Suresh S, Mohmood R, Ramachandra YL, Pandey A: **An evaluation of human protein-protein interaction data in the public domain.** *BMC Bioinformatics* 2006, **7 Suppl 5**:S19.
28. Dempster AP: **Upper and Lower Probabilities Induced by a Multivalued Mapping.** *Ann Math Stat* 1967, **38**:325-&.
29. Reich M, Liefeld T, Gould J, Lerner J, Tamayo P, Mesirov JP: **GenePattern 2.0.** *Nat Genet* 2006, **38**:500-501.
30. Fraser HB, Hirsh AE, Wall DP, Eisen MB: **Coevolution of gene expression among interacting proteins.** *Proceedings of the National Academy of Sciences of the United States of America* 2004, **101**:9033-9038.
31. Mintseris J, Weng Z: **Structure, function, and evolution of transient and obligate protein-protein interactions.** *Proc Natl Acad Sci U S A* 2005, **102**:10930-10935.
32. Tsai CJ, Xu D, Nussinov R: **Protein folding via binding and vice versa.** *Fold Des* 1998, **3**:R71-80.
33. Jones S, Thornton JM: **Principles of protein-protein interactions.** *Proc Natl Acad Sci U S A* 1996, **93**:13-20.

34. Coates PJ, Hall PA: **The yeast two-hybrid system for identifying protein-protein interactions.** *J Pathol* 2003, **199**:4-7.
35. Reguly T, Breitkreutz A, Boucher L, Breitkreutz BJ, Hon GC, Myers CL, Parsons A, Friesen H, Oughtred R, Tong A, et al: **Comprehensive curation and analysis of global interaction networks in *Saccharomyces cerevisiae*.** *J Biol* 2006, **5**:11.
36. Ewing RM, Chu P, Elisma F, Li H, Taylor P, Climie S, McBroom-Cerajewski L, Robinson MD, O'Connor L, Li M, et al: **Large-scale mapping of human protein-protein interactions by mass spectrometry.** *Mol Syst Biol* 2007, **3**:89.
37. Zuiderweg ER: **Mapping protein-protein interactions in solution by NMR spectroscopy.** *Biochemistry* 2002, **41**:1-7.
38. Jansen R, Greenbaum D, Gerstein M: **Relating whole-genome expression data with protein-protein interactions.** *Genome Res* 2002, **12**:37-46.
39. Zhang G, Campbell EA, Minakhin L, Richter C, Severinov K, Darst SA: **Crystal structure of *Thermus aquaticus* core RNA polymerase at 3.3 Å resolution.** *Cell* 1999, **98**:811-824.
40. Aytuna AS, Gursoy A, Keskin O: **Prediction of protein-protein interactions by combining structure and sequence conservation in protein interfaces.** *Bioinformatics* 2005, **21**:2850-2855.
41. Ogmen U, Keskin O, Aytuna AS, Nussinov R, Gursoy A: **PRISM: protein interactions by structural matching.** *Nucleic Acids Res* 2005, **33**:W331-336.
42. Jansen R, Yu H, Greenbaum D, Kluger Y, Krogan NJ, Chung S, Emili A, Snyder M, Greenblatt JF, Gerstein M: **A Bayesian Networks Approach for Predicting Protein-Protein Interactions from Genomic Data.** *Science* 2003, **302**:449-453.
43. Qi Y, Bar-Joseph Z, Klein-Seetharaman J: **Evaluation of different biological data and computational classification methods for use in protein interaction prediction.** *Proteins* 2006, **63**:490-500.
44. Lin N, Wu B, Jansen R, Gerstein M, Zhao H: **Information assessment on predicting protein-protein interactions.** *BMC Bioinformatics* 2004, **5**:154.
45. Shlomi T, Segal D, Ruppin E, Sharan R: **QPath: a method for querying pathways in a protein-protein interaction network.** *BMC Bioinformatics* 2006, **7**:199.
46. Hermjakob H, Montecchi-Palazzi L, Lewington C, Mudali S, Kerrien S, Orchard S, Vingron M, Roechert B, Roepstorff P, Valencia A, et al: **IntAct: an open source molecular interaction database.** *Nucleic Acids Res* 2004, **32**:D452-455.
47. Xenarios I, Salwinski L, Duan XJ, Higney P, Kim SM, Eisenberg D: **DIP, the Database of Interacting Proteins: a research tool for studying cellular networks of protein interactions.** *Nucleic Acids Res* 2002, **30**:303-305.
48. Alfarano C, Andrade CE, Anthony K, Bahroos N, Bajec M, Bantoft K, Betel D, Bobechko B, Boutilier K, Burgess E, et al: **The Biomolecular Interaction Network Database and related tools 2005 update.** *Nucleic Acids Res* 2005, **33**:D418-424.
49. Maglott D, Ostell J, Pruitt KD, Tatusova T: **Entrez Gene: gene-centered information at NCBI.** *Nucleic Acids Res* 2007, **35**:D26-31.
50. Rhodes DR, Tomlins SA, Varambally S, Mahavisno V, Barrette T, Kalyana-Sundaram S, Ghosh D, Pandey A, Chinnaiyan AM: **Probabilistic model of the human protein-protein interaction network.** *Nat Biotechnol* 2005, **23**:951-959.
51. Schwikowski B, Uetz P, Fields S: **A network of protein-protein interactions in yeast.** *Nat Biotechnol* 2000, **18**:1257-1261.

52. Overbeek R, Fonstein M, D'Souza M, Pusch GD, Maltsev N: **The use of gene clusters to infer functional coupling.** *Proc Natl Acad Sci U S A* 1999, **96**:2896-2901.
53. Jothi R, Cherukuri PF, Tasneem A, Przytycka TM: **Co-evolutionary analysis of domains in interacting proteins reveals insights into domain-domain interactions mediating protein-protein interactions.** *J Mol Biol* 2006, **362**:861-875.
54. Doderer M, Yoon K, Salinas J, Kwek S: **Protein subcellular localization prediction using a hybrid of similarity search and error-correcting output code techniques that produces interpretable results.** *In Silico Biol* 2006, **6**:419-433.
55. Doderer M, Yoon K, Kwek S: **Species Independent Protein Localization Prediction for Multi-compartmentalized Proteins.** In *The 2007 International Conference on Machine Learning: Models, Technologies & Applications; Las Vegas, NV.*; 2007
56. Lee K, Kim DW, Na D, Lee KH, Lee D: **PLPD: reliable protein localization prediction from imbalanced and overlapped datasets.** *Nucleic Acids Res* 2006, **34**:4655-4666.
57. Karmaker A, Doderer M, Harris SE, Kwek S: **Discovery of Transcription Factors Using Protein Subcellular Localization Prediction and Gene Expression Profile Analysis.** In *The 2007 International Conference on Machine Learning: Models, Technologies & Applications; Las Vegas, NV.*; 2007
58. Altschul SF, Gish W, Miller W, Myers EW, Lipman DJ: **Basic local alignment search tool.** *J Mol Biol* 1990, **215**:403-410.
59. Frank E, Hall M, Trigg L, Holmes G, Witten IH: **Data mining in bioinformatics using Weka.** *Bioinformatics* 2004, **20**:2479-2481.
60. Feng ZP, Zhang CT: **Prediction of the subcellular location of prokaryotic proteins based on the hydrophobicity index of amino acids.** *Int J Biol Macromol* 2001, **28**:255-261.
61. Guo J, Lin Y, Sun Z: **A novel method for protein subcellular localization: combining residue-couple model and svm.** In *Proceedings of 3rd Asia-Pacific Bioinformatics Conference; Singapore.* 2005
62. Barnes MR: *Bioinformatics for geneticists : a bioinformatics primer for the analysis of genetic data.* 2nd edn. Chichester, England ; Hoboken, NJ: Wiley; 2007.
63. Gewehr JE, Szugat M, Zimmer R: **BioWeka--extending the Weka framework for bioinformatics.** *Bioinformatics* 2007, **23**:651-653.
64. Huh WK, Falvo JV, Gerke LC, Carroll AS, Howson RW, Weissman JS, O'Shea EK: **Global analysis of protein localization in budding yeast.** *Nature* 2003, **425**:686-691.
65. Salwinski L, Miller CS, Smith AJ, Pettit FK, Bowie JU, Eisenberg D: **The Database of Interacting Proteins: 2004 update.** *Nucleic Acids Res* 2004, **32**:D449-451.
66. Peri S, Navarro JD, Amanchy R, Kristiansen TZ, Jonnalagadda CK, Surendranath V, Niranjana V, Muthusamy B, Gandhi TK, Gronborg M, et al: **Development of human protein reference database as an initial platform for approaching systems biology in humans.** *Genome Res* 2003, **13**:2363-2371.
67. Zanzoni A, Montecchi-Palazzi L, Quondam M, Ausiello G, Helmer-Citterich M, Cesareni G: **MINT: a Molecular INTeraction database.** *FEBS Lett* 2002, **513**:135-140.
68. Pagel P, Kovac S, Oesterheld M, Brauner B, Dunger-Kaltenbach I, Frishman G, Montrone C, Mark P, Stumpflen V, Mewes HW, et al: **The MIPS mammalian protein-protein interaction database.** *Bioinformatics* 2005, **21**:832-834.

69. Beuming T, Skrabanek L, Niv MY, Mukherjee P, Weinstein H: **PDZBase: a protein-protein interaction database for PDZ-domains.** *Bioinformatics* 2005, **21**:827-828.
70. Joshi-Tope G, Gillespie M, Vastrik I, D'Eustachio P, Schmidt E, de Bono B, Jassal B, Gopinath GR, Wu GR, Matthews L, et al: **Reactome: a knowledgebase of biological pathways.** *Nucleic Acids Res* 2005, **33**:D428-432.
71. Brown KR, Jurisica I: **Online predicted human interaction database.** *Bioinformatics* 2005, **21**:2076-2082.
72. Stark C, Breitkreutz BJ, Reguly T, Boucher L, Breitkreutz A, Tyers M: **BioGRID: a general repository for interaction datasets.** *Nucleic Acids Res* 2006, **34**:D535-539.
73. Han JD, Bertin N, Hao T, Goldberg DS, Berriz GF, Zhang LV, Dupuy D, Walhout AJ, Cusick ME, Roth FP, Vidal M: **Evidence for dynamically organized modularity in the yeast protein-protein interaction network.** *Nature* 2004, **430**:88-93.
74. Stelzl U, Worm U, Lalowski M, Haenig C, Brembeck FH, Goehler H, Stroedicke M, Zenkner M, Schoenherr A, Koeppen S, et al: **A human protein-protein interaction network: a resource for annotating the proteome.** *Cell* 2005, **122**:957-968.
75. Grossmann S, Bauer S, Robinson PN, Vingron M: **Improved detection of overrepresentation of Gene-Ontology annotations with parent child analysis.** *Bioinformatics* 2007, **23**:3024-3031.
76. Parkinson H, Kapushesky M, Kolesnikov N, Rustici G, Shojatalab M, Abeygunawardena N, Berube H, Dylag M, Emam I, Farne A, et al: **ArrayExpress update--from an archive of functional genomics experiments to the atlas of gene expression.** *Nucleic Acids Res* 2009, **37**:D868-872.
77. Goodman SN: **Toward evidence-based medical statistics. 1: The P value fallacy.** *Ann Intern Med* 1999, **130**:995-1004.
78. Goodman J, Yih W-t: **Online Discriminative Spam Filter Training.** In *CEAS 2006 - Third Conference on Email and Anti-Spam; July 27-28, 2006; Mountain View, California USA.* 2006
79. Dempster AP, Laird NM, Rubin DB: **Maximum Likelihood from Incomplete Data Via Em Algorithm.** *J Roy Stat Soc B Met* 1977, **39**:1-38.
80. Cover T, Hart P: **Nearest neighbor pattern classification.** *IEEE Transactions on Information Theory* 1967, **13**:21-27.

VITA

Mark Doderer was born in Colorado Springs, Colorado and was raised mostly in San Antonio Texas. He graduated from Marshall High School in San Antonio in 1986. He graduated from Trinity University with a Bachelor's degree in Computer Science in 1990. After completing the secondary education certification program at University of Texas at Dallas he taught high school Computer Science and junior high Computer Literacy. After teaching for five years he started a small web site consulting company with his brother and enjoyed that challenge until he started his graduate work at University of Texas at San Antonio. He completed his Master's Degree in Computer Science in 2006 and his Doctor of Philosophy in Computer Science in 2010.

Besides taking courses and pursuing research in machine learning and bioinformatics, he taught several of the computer science courses at the University of Texas at San Antonio, including Cultural Implications of the Information Society, Microcomputer Applications, Introduction to Computer Programming, Data Structures and Advanced Programming.

He looks forward to continuing research in bioinformatics as a member of the faculty at the University of Texas Health Science Center of San Antonio. He would also enjoy teaching undergraduate Computer Science courses if the opportunity arises.

Mark is married to Marcy Doderer and has two daughters Emily age 15 and Katherine age 12. Mark's parents are Sylvia Doderer and Earl Doderer Ph.D.