The Gene Ontology (GO) provides a terminology to describe biological concepts and their relationships in a consistent and species-independent manner. Recent research has begun exploring the use of formal concept analysis (FCA) on annotated biological objects such as genes and gene products to discover the similarity and relationships among them. This thesis has produced a generic FCA software tool that creates a variety of concept lattices and incorporates the structure of the ontological terminology used for annotations. The user can tailor the creation of the concept lattices based on characteristics of the annotations and view a 3D visualization of the results. Numerous querying capabilities for exploring the resulting concept lattices are also included. Real-world data including a breast-cancer gene list from the University of Cincinnati’s Children’ Hospital and Medical Center and a well-known set of gene products GPD194 from the research literature are used to evaluate the FCA software.
CONCEPT LATTICE ANALYSIS FOR ANNOTATED OBJECTS

A Thesis

Submitted to the
Faculty of Miami University
in partial fulfillment of
the requirements for the degree of
Master in Computer Science
Department of Computer Science and System Analysis
by
Wenting Yi
Miami University
Oxford, Ohio
2009

Advisor______________________________________
Valerie Cross, PhD.

Reader_______________________________________
Alton Sanders, PhD.

Reader_______________________________________
Michael Zmuda, PhD.
## TABLE OF CONTENTS

1 Introduction ........................................................................................................... 1
2 Ontological Terminologies ..................................................................................... 2
  2.1 Gene Ontology .................................................................................................. 4
  2.2 Annotation ....................................................................................................... 5
3 Background Knowledge ......................................................................................... 6
4 Related Research .................................................................................................. 10
  4.1 QUOTA ............................................................................................................ 11
  4.2 Algorithms for Building Concept Lattices ..................................................... 12
  4.3 Combining Rough Set Theory and Concept Lattices ...................................... 17
  4.4 Applying Concept Lattice Representations in Bioinformatics ....................... 20
  4.5 Concept Approximation Theory ..................................................................... 22
    4.5.1 Rough Set Theory Concept Approximation ........................................... 23
    4.5.2 Concept Approximation using Irreducibility ......................................... 24
    4.5.3 Concept Approximation using Lattice-Theoretic Operators ................... 25
  4.6 Similarity Measures between Concepts in a Concept Lattice ......................... 26
    4.6.1 A Set-theoretic Basic .............................................................................. 26
    4.6.2 Concept Similarity ................................................................................... 27
  4.7 Assessing Concept Lattice Similarity ............................................................. 29
5 Concept Lattice Creation and Querying Subsystem of QUOTA ....................... 31
  5.1 Concept Lattice Creation ............................................................................... 32
  5.2 Filtered and Granulated Concept Lattice Construction .................................... 35
    5.2.1 Constraining the Selection of Annotations ............................................ 36
    5.2.2 Granulated Concept Lattice Algorithm ................................................. 38
  5.3 Concept Approximation Querying ................................................................. 47
  5.4 Concept Similarity Assessment ...................................................................... 47
  5.5 Concept Lattice Similarity Assessment .......................................................... 49
6 User Interface ....................................................................................................... 50
  6.1 Concept Lattice Panel ..................................................................................... 51
  6.2 Granulation and Filtering Panel ...................................................................... 54
  6.3 CL Similarity Panel ....................................................................................... 56
  6.4 Query Panel ................................................................................................... 58
7 Evaluation ............................................................................................................ 59
  7.1 Variety of Concept Lattice .............................................................................. 60
  7.2 Granulation and Filtering of Concept Lattice ................................................. 63
  7.3 Concept Lattice Similarity ............................................................................. 74
  7.4 Concept Similarity ........................................................................................ 84
  7.5 Concept Approximation ............................................................................... 88
8 Research Significance and Future Research Opportunities .......................... 92
9 Reference ............................................................................................................ 95
10 Appendix ........................................................................................................... 98
LIST OF TABLES

Table 7-1 Similarity Results for Original CL and Granulated CLs of the Training Set………72
Table 7-2 Characteristics of the GPD194 Data Set [Popescu et al. 2006]…………………….75
Table 7-3 Concept Lattice Similarity Results of GDP194 Data………………………………76
Table 7-4 Rank of Potential Breast Cancer Causing Genes Found in the Test Set……………86
Table 7-5 Comparison of the Similarity Results using GIC and C-SS-Tversky……………….87
Table 7-6 A Formal Context Example in Saquer’s Paper...................................................89
Table 7-7 Result of Extents of LA and UA Concepts of Object Sets…………………………91
LIST OF FIGURES

Figure 2-1 Example of GO (http://www.ebi.ac.uk/ego) ..........................................................4
Figure 2-2 Example of GO with Associated Annotations (Kaiser et al, 2006) ......................6
Figure 3-1 A Formal Context Example and Its Concept Lattices (Yao, 2004) ......................7
Figure 4-1 Concept Lattice Shown in (Kaiser et al. 2006) from Using the Annotations in GO BP
Portion Shown in Figure 2.2 ........................................................................................................21
Figure 5-1 First Step of Algorithm to Create Granulated Concept Lattice .........................40
Figure 5-2 Second Step of Algorithm to Create Granulated Concept Lattice ....................41
Figure 5-3 Third Step of Algorithm to Create Granulated Concept Lattice .....................43
Figure 5-4 Fourth Step of Algorithm to Create Granulated Concept Lattice ....................44
Figure 5-5 Fifth Step of Algorithm to Create Granulated Concept Lattice .......................46
Figure 6-1 Overall User Interface of QUOTA ........................................................................51
Figure 6-2 User Interface of Concept Lattice Panel .............................................................52
Figure 6-3 Result of a Concept Lattice .................................................................................54
Figure 6-4 User Interface of Granulation & Filtering Panel ..................................................56
Figure 6-5 User Interface of Concept Lattice Similarity Panel .............................................57
Figure 6-6 User Interface of Query Panel .............................................................................59
Figure 7-1 Result of General Concept Lattice Given by the Software ................................61
Figure 7-2 Result of OOCL Given by the Software ...............................................................62
Figure 7-3 Result of Concept POCL by the Software ............................................................62
Figure 7-4 Result of Complement Concept Lattice Given by the Software .........................63
Figure 7-5 Concept Lattice of Figure 2-2 ............................................................................64
Figure 7-6 List of Concepts of Figure 7-5 ............................................................................65
Figure 7-7 Granulated Concept Lattice on Levels of Figure 2-2 .........................................66
Figure 7-8 List of Concepts of Figure 7-7 ............................................................................67
Figure 7-9 Concept Lattice of Training Set .........................................................................69
Figure 7-10 Concept Lattice of Training Set Granulated to Level 6 .....................................70
Figure 7-11 Concept Lattice of Training Set Granulated to Level 3 .....................................71
Figure 7-12 Granulated Concept Lattice on IC/IR Values of Figure 2-2 ..............................73
Figure 7-13 List of Concepts of Figure 7-12 .......................................................................73
Figure 7-14 Equivalence Classes of the Granulated CL in Figure 7-12 ...............................74
ACKNOWLEDGEMENT

I would like to give my sincere thanks and appreciation to Prof. Valerie Cross for her valuable advice, constant encouragement and support. Without her guidance, I could not have finished this work. I also want to thank my thesis committee members, Prof. Alton Sanders and Prof. Michael Zmuda for their guidance and support.

I am also grateful to Sriram Ramakrishnan for his help in providing the GO semantic similarity results that have been using in my research, and thank Catia Pesquita in providing the example data I used in this thesis.

Finally, I want to thank my husband En Yu for his support, help and continuous encouragement.
1 Introduction

Today, large databases are built to describe genomic information, relationships between genes and large amount of analyzed data due to the increasing development of computational biology. To guarantee the consistency of biological concepts used in different databases, the Gene Ontology (GO) (Gene Ontology Consortium, 2000) has been developed to describe biological concepts and their relationships in a species-independent manner. A variety of software has been developed to assess the similarity of genes and gene products based on the similarity of their annotations. Just recently researchers have begun to explore using formal concept analysis (FCA) on annotated objects in order to analyze the similarity or discover the relationship among the annotated objects (Kaiser et al, 2006; Hu et al, 2007; Choi et al 2006).

Building the concept lattice of a binary relation has often been used in the field of knowledge discovery from large databases. Wille (1982) proposed to consider each element in the lattice as concept and the corresponding graph (Hasse diagram) as the generalization/specialization relationship between concepts. With this interpretation the lattice represents a concept hierarchy. Many algorithms have been proposed for generating the concept lattice from a binary relation. But very little effort has been devoted to building formal concepts on the basis of a set of annotated objects and an associated ontological terminology that organizes the annotation terms. The ontological terms and the annotated objects can be viewed as elements of the two required sets in FCA. The difficulty of using formal concept analysis in an ontological terminology is that the terms are organized in a generalization/specialization hierarchy; they are not indivisible properties attributed to an annotated object. For example, if an object is annotated with term $t$ which is a specialization of term $s$ then the object is also assumed to be annotated with term $s$.

This thesis research investigates approaches to building concept lattices using the structure of the ontological terminology and sets of annotated objects using terms from an ontological terminology. In particular, this research makes the following contributions: 1) a user friendly
and generic FCA tool that allows a variety of approaches to create concept lattices from annotation files, 2) filtering and granulation of concept lattices based on the structure of the ontological terminology, 3) 3D visualization of the concept lattices, 4) several methods for concept approximation including both lower and upper approximations, 5) concept similarity measures between formal concepts, 6) a method of assessing overall similarity between two concept lattices, 7) additional querying capabilities on concept lattices, and 8) an evaluation of the features of this FCA tool using real world data. This FCA analysis tool is integrated with the annotation querying system QUOTA (Sun 2007).

The description of this thesis work is organized as follows. Chapter 2 provides a brief introduction to ontological terminologies and their use in annotating objects; in particular the Gene Ontology and annotation files are used to illustrate the important ideas necessary for this thesis research. In Chapter 3 important definitions and mathematical notation used in FCA are overviewed. Chapter 4 first briefly describes the QUOTA system and then presents recent research in concept lattice building algorithms and presents the foundation research for all of the analysis features implemented in the QUOTA FCA software. Chapter 5 provides a discussion of the implementation of these features. The user interface for the FCA software is presented in Chapter 6. The evaluation and validation of the software is described in Chapter 7. Finally Chapter 8 examines the significance of this thesis research and opportunities for future research from this thesis work.

2 Ontological Terminologies

An ontology often contains the following elements:

1. Instances: A specific object or anything that can be described by the concepts in an ontology.
2. Concepts: Abstract types of instances
3. Attributes: The properties or characteristics any concepts can have and share.
4. Relations: how the concepts can be related to each other.

An ontology also serves as a vocabulary that represents a set of concepts of a particular domain and the relationships between those concepts. An ontological terminology is somewhat different from more general ontologies in that the instances are the actual concepts labeled by words or terms in the domain vocabulary. The main concept (meta concept in the intensional or definitional ontology) is a ‘concept’, for example in WordNet ontology, the concept ‘lexical concept’ is defined and instances of lexical concepts such as ‘vehicle’ or ‘car’ are created. These instances of lexical concepts can be used to annotate other objects or instances from another domain.

The core structure of an ontology is its multi-inheritance hierarchy. For example, the generalization/specialization hierarchy, also referred to as the broader/narrower than hierarchy in ontological terminologies, is the backbone structure. For example, the instance of the concept vehicle would be broader than the instance of the concept car. In the object-oriented field, the vehicle class would be a superclass of car and car would be a subclass of vehicle. In other words, the class car would inherit the attributes belonging to the class vehicle; moreover car itself would have additional attributes specific to itself.

The main difference between a general ontology and an ontological terminology is what is considered the intensional part and what is consider the extensional part. The intensional part is the definitional part of the ontology. For an ontological terminology, the intensional part typically resembles the components of a thesaurus and consists of classes such as lexicalConcepts, nouns, verbs, word objects and so on and relationships such as broader than/narrower than, synonymFor/hasSynonyms, antonymOf/has Antonyms and so on. The extensional part consists of the actual instances of lexical concepts such as car and vehicle. In a general ontology, the lexical concepts such as car and vehicle become the classes and the instances are actual real world objects that are cars that have other attributes such as data of purchase, current mileage and so on.
2.1 Gene Ontology (GO)

The Gene Ontology, or GO, developed by Gene Ontology Consortium, is used to annotate genes and gene products. In GO, there are three sub-ontologies which represent the three key orthogonal areas that a gene or gene product can be related to:

1. **Cellular Component**: It consists of all the known parts that any cells may have.
2. **Biological Process**: It consists of a processes that are a series of events accomplished by one or more ordered assemblies of molecular functions.
3. **Molecular Function**: It consists of biological activities that occur at the molecular level.

In addition to these three major sub-ontologies, GO is structured by only two relationships and their inverses: IS_A and PART_OF. The IS_A relationship constructs the taxonomic or broader than/narrower than hierarchy of GO. The PART_OF relationship is the same as meronym semantic relation in linguistics which designates a basic component, part or member of something. GO only has one kind of PART_OF relationship while WordNET had six different kinds. For example, the nucleus is PART_OF the cell. Below is a figure illustrating some of the in the hierarchy of the Biological Process subontology of the GO.

![Figure 2-1 Example of GO](http://www.ebi.ac.uk/ego)
2.2 Annotations

An annotation typically represents some information that describes something in more detail. In Gene Ontology, an annotation represents a mapping between the concepts in the GO and the actual genes and gene products described in biological databases. It can also be viewed as a process of further describing genes and gene products and may be used to establish relationships between them since annotation with a GO term, also connects the gene or gene product to other terms in the GO.

Since GO is an open source project, users can download GO annotation files from GO annotation web site: http://www.geneontology.org/GO.current.annotations.shtml. It contains all GO annotation files submitted by the GO Consortium members. There are typically two files, filtered and unfiltered files. Filtered files are those have been filtered by the annotation file QC checks script and are non-redundant files. Unfiltered files are those that have not yet been checked for redundancy.

To create a standard GO annotation file, GO Consortium sets its own restrictions on the annotation file format. “It uses a tab-delimited format to represent annotation data. Each line of the annotation data represents a single link between a gene product and a GO concept with a certain evidence code and the reference to support the link.” (GO Annotation File Format Guide). The following figure illustrates part of the GO ontology with example genes or gene products associated with the GO concepts.
3 Background Knowledge

Formal concept analysis, introduced by Rudolf Wille and his students, is a method of data analysis that takes an input matrix specifying a set of objects and their associated properties and finds both all the "natural" clusters of properties and all the "natural" clusters of objects in the input data. (http://en.wikipedia.org/wiki/Formal_concept_analysis) This section reviews many basis yet important definitions of FCA theory.

Definition 1: A formal context $K := (G, M, I)$ consists of two set $G$ and $M$ and a relation $I$ between $G$ and $M$. The elements of $G$ are called the objects and the elements of $M$ are called the attributes of the context. In order to express that an object $g$ is in a relation $I$ with an attribute $m$, we write $g I m$ or $(g, m) \in I$, and read it as “the object $g$ has the attribute $m$”. (Qiao et al., 2003)
Figure 3-1 A Formal Context Example and Its Concept Lattices (Yao, 2004)

For example, in figure 3-1, the context is represented by a cross table. It represents a formal context $K = (G, M, I)$, where $G = \{1, 2, 3, 4, 5, 6\}$ and $M = \{a, b, c, d, e\}$. A cross in row $g \in G$ and column $m \in M$ means that the object $g$ has the attribute $m$, or $g$ and $m$ have the relation $I$.

Other terminology is sometimes used for sets $G$ and $M$. Set $G$ may be referred to as a set of transactions and set $M$ as a set of items.

**Definition 2:** For a set $A \subseteq G$ of objects we define $A' := \{m \in M \mid gI_m \text{ for all } g \in A\}$ (the set of attributes common to [all] the objects in $A$). Correspondingly, for a set $B$ of attributes we define $B' := \{g \in G \mid gI_m \text{ for all } m \in B\}$ (the set of objects which have all attributes in $B$). (Qiao et al., 2003)

In the above example, if $A = \{1, 2, 5\}$, then $A' = \{a\}$. If $B = \{a, c\}$, then $B' = \{1, 2\}$.

**Definition 3a:** Let $gI = \{m \in M \mid gI_m\}$, the set of attributes possessed by object $g$. Define the relation $R$ on $G$ as follows $g_1Rg_2$ iff $g_1I = g_2I$ where $g_1, g_2 \in G$. $R$ is an equivalence relation on
G and induces a partition on G. Let G/R be the set of all object equivalence classes induced by
R on G.

The extent of each bottom level concept (parents of the bottom concept which has an empty
extent) represents an object equivalence class; however, more object equivalence classes may
exist besides those found in the bottom level concepts.

Definition 3b: Let \( Im = \{ g \in G \mid gIm \} \), the set of objects possessing attribute m. Define the
relation \( R' \) on M as follows \( m_1Rm_2 \) iff \( Im_1 = Im_2 \) where \( m_1, m_2 \in M \). \( R' \) is an equivalence
relation on M and induces a partition on M. Let \( M/R' \) be the set of all attribute equivalence
classes induced by \( R' \) on M.

The intent of each top level concept (children of the top concept which has an empty intent)
represents an attribute equivalence class; however, more attribute equivalence classes may exist
besides those found in the top level concepts.

Definition 4: A formal concept of the context \((G, M, I)\) is a pair \((A, B)\) with \( A \subseteq G, B \subseteq M, A' = B \) and \( B' = A \). We call A the extent and B the intent of concept \((A, B)\). \( B(G, M, I) \) denotes the set of all concepts of the context \((G, M, I)\). (Qiao et al. 2003)

The intent of a formal concept is also referred to as a closed itemset. This terminology is used
when set G is considered a set of items.

A concept lattice is made of several formal concepts with each having its extent and intent. From
the graph above, the concept lattice in figure 3.1 which is labeled CL and is below the formal
context table has 9 concepts. The extent of the top formal concept contains \{1, 2, 3, 4, 5, 6\} and
its intent is an empty set indicating that these 6 objects in the extent do not have a single
common attribute. The notation $B(G, M, I)$ represents all 9 formal concepts in this concept lattice: \{(12345, $\emptyset$), (1256, a), (1346, e), (12, ac), (16, ae), (346, be), (1, acde), (6, abe), ($\emptyset$, abcde)\}. In concept lattice, when two concepts are connected by an edge, the extent (intent) of the concept higher in the concept lattice is a superset (subset) of the extent (intent) of the other connected concept. No other formal concepts can be placed between these two connected formal concepts.

**Definition 5:** In the concept lattice, the meet of two or more concepts is the concepts’ greatest-lower-bound (nearest common descendant), while the join of two or more concepts is the concepts’ least-upper-bound (nearest common ancestor).

For example, in the CL on the left in Figure 3-1, concepts (1, acde) and (6, abe) have the join of (16, ae) and their meet is ($\emptyset$, abcde). The join of (1, acde) and (346, be) is (1346, e).

**Definition 5:** In the concept lattice, the meet-irreducible node is the node that only has one parent so that it can not be written as a meet of any two nodes, while the join-irreducible node is the node that only has one child so that it can not be written as a join of any two nodes.

For example, in the OOCL in Figure 3-1, node (12, cd) is join-irreducible, but is not meet-irreducible since it can be written as the meet of node (1256, acd) and (12346, bcde).

**Definition 6:** Let $K := (G, M, I)$ be a formal context, for an object $g \in G$, we write $g'$ instead of $\{g\}'$ for the object intent $\{m \in M \mid gIm\}$ of the object $g$, and also $m'$ instead of $\{m\}'$ for the attribute extent $\{g \in G \mid gIm\}$ of the attribute $m$. A basis $B$ is the set of all attribute extents of $K$, i.e., $B = \{m' \mid m \in M\}$. Also we denote by $F_B$ the family generated by join from the basis $B$, i.e., $F_B = \{\bigcap_{m' \in I} m' \mid I \in 2^B\}$. For each $F \subseteq F_B$, we denote by $\gamma(F)$...
the subset of \( M \), such that for each element \( m \) in it, \( F \) is included in \( m' \), i.e., \( \gamma(F) = \{ m \in M \mid F \subseteq m' \} \).

In the concept lattice in figure 3-1, the basis \( B = \{a' = \{1256\}, b' = \{346\}, c' = \{12\}, d' = \{1\}, e' = \{1346\} \}, \) the family \( F_B = \{\{123456\}, \{1256\}, \{346\}, \{12\}, \{1\}, \{1346\}, \{6\}, \{16\}, \emptyset \} \), and \( \gamma(F) = \{ \emptyset, \{a\}, \{be\}, \{ac\}, \{acde\}, \{e\}, \{abe\}, \{ae\}, \{abcde\} \} \).

The definition of the object oriented concept lattice (OOCL) and the property oriented concept lattice (POCL) are defined more formally in section 4.3 with the introduction of approximation operators using rough set theory. Here an informal definition is given and examples labeled OOCL in the middle of figure 3.1 and labeled POCL on the right of Figure 3.1 are provided. For an OOCL, the intent of the formal concept specifies a set of attributes, each of which map to a subset of the objects in the extent of the OOCL formal concept. For a POCL, the extent of the formal concept specifies a set of objects, each of which map to a subset of the attributes in the intent of the POCL formal concept.

### 4 Related Research

In this section, various papers found in the research literature are summarized and their importance to this thesis research outlined. Some present algorithms needed to pursue this thesis research. Others are just beginning to pursue the same research that this thesis is proposing, i.e., investigating how to apply concept lattice theory to analyze or better understand relationships and similarities between annotated objects. Some papers, however, present only ideas and definitions but have not provided the algorithms. This thesis research has developed and extended the needed algorithms to build an integrated set of features for the resulting FCA software which has been incorporated into the QUOTA system developed as part of a thesis.
4.1 QUOTA

QUOTA (Sun 2007) is a generic tool developed to process a variety of queries that use an annotating ontology and a set of objects annotated by terms from that ontology. Currently there are several different existing queries using different kinds of input and producing different kinds of output. The current queries consist of:

1. **Set of Annotated Objects IN and Ontological Terms per Object OUT**
   This query lists each object in the set with its corresponding ontological terms.

2. **Set of Annotated Objects IN and Ranked Ontological Terms for Set OUT**
   This query produce a set of ontological terms that best categorizes or summarizes the input set of objects using the ontology of annotating terms.

3. **Set of Annotated Objects IN and a Similarity Matrix between Objects OUT**
   This query produces the similarity between each pair of annotated objects in the input set based on the semantic similarity among their annotating terms. Various semantic similarity measures may be used to determine matrix values. The structure of the ontology of annotating terms is used in determining the semantic similarity.

4. **Two Objects each described by a set of annotated objects IN and Semantic Similarity between the two objects OUT**
   This query produces the similarity score between the two input objects based on the similarity between the two sets of annotated objects. It produces a higher level similarity based on the similarity between the annotated objects.

This thesis research enhances querying on annotated objects by adding formal concept analysis capabilities to QUOTA. The annotating terms from the ontological terminology correspond to
the attributes of a formal context, i.e. the set M, and the annotated objects correspond to the objects of a formal context, i.e., the set G. The implementation of this FCA query is to provide several features for analysis of set of annotated objects that extend and expand the QUOTA system.

### 4.2 Algorithms for Building Concept Lattices

Numerous papers exist in the research literatures that present algorithms for build concept lattices. The performances of several of these have been compared both experimentally and theoretically \((\text{Kuznetsov 2002})\). The conclusion of that research states that choosing an algorithm depends on the properties of the input data and provides several recommendations.

Several proposed algorithms that appear in the literature after that comparison suggest in their paper titles that they are more efficient or fast algorithms. In this section, two of these algorithms are reviewed and the similarity and differences between them are discussed.

In \((\text{Qiao et al. 2003})\) an efficient algorithm is presented that first computes the basis of the context and uses the basis to derive all the formal concepts. Then from a theorem proved in the paper, an algorithm is developed that can easily compute the cover relation of the concepts, i.e., the concept lattice structure which links the formal concepts. Qiao provides three algorithms that can be used to create a concept lattice from a given formal context.

Algorithm 1 computes the basis \(B\) of the formal context table. The input of the algorithm is the formal context, and the output is the basis \(B\).

```plaintext
begin
#initialize B
let \(|B| = |M|\)
```
for each \( m' \in B \) do
\[
m' = \emptyset
\]

for each \( m' \in B \) do

for each \( g \in G \) do

if \( g \upharpoonright m \) then \( m' = m' \cup \{g\} \)

end

Using the concept lattice in figure 3.1 as an example, algorithm 1 can compute the basis \( B = \{a' = \{1, 2, 5, 6\}, b' = \{3, 4, 6\}, c' = \{1, 2\}, d' = \{1\}, E' = \{1, 3, 4, 6\}\}. There are 5 elements in this \( B \), and each element is represented as an \( m' \) in the algorithm.

Algorithm 2 computes the extents and intents of each one in the family so that every concept in the concept lattice can be generated. The input is the basis \( B \), and the output is the concept family \( F_B \).

begin

#initialize \( F_B \)

let \( F_B = \{G\}, \gamma(G) = \emptyset \)

for each \( m' \in B \) do

if \( m' = G \) then \( \gamma(G) = \gamma(G) \cup \{m\} \)

for each \( m' \in B \) do

for each \( F \subseteq F_B \) do

begin

\( F' = F \cap m' \)

if \( F' \not\in F_B \) then

begin

\( F_B = F_B \cup F' \)

end

end

end

end
Algorithm 2 continues to compute each formal concept in the lattice. If we use \( a' = \{1, 2, 5, 6\} \) in the above example as an \( m' \) here, we can follow the steps and first set \( F_B = \{G\} = \{\{1, 2, 3, 4, 5, 6\}\} \) and \( \gamma(G) = \emptyset \). Then it checks whether \( \{1, 2, 5, 6\} \) equals \( G \) or not. The result is false, so it proceeds to the next for loop. Doing the intersection of the only element in \( F_B \) and \( m' \) produces \( F' = m' = \{1, 2, 5, 6\} \) which is not in \( F_B \); therefore, this set is added as an element to \( F_B \). Finally let \( \gamma(F') = \gamma(\{1, 2, 5, 6\}) = \emptyset \cap \{m'\} = \{a\} \). Now we get one concept in the lattice: \((1256, a)\).

Algorithm 3 introduces how to construct the concept lattice, that is, how to add edges between concepts and how to decide the concept lattice hierarchy. The input is the concept family \( F_B \), and the output is the final concept lattice.

begin
#initialize
for each \( F \in F_B \) do
\( \text{COUNT}(F) = 0 \)
for each \( F \in F_B \) do
    for each \( m' \in B \mid \gamma(F) \) do
        begin
            \( F' = F \cap m' \)
            \( \text{COUNT}(F')++ \)
            if \( |\gamma(F')| = \text{COUNT}(F') + |\gamma(F)| \) then
                add an edge from \((F, \gamma(F))\) to \((F', \gamma(F'))\)
end
In algorithm 2, all 9 concepts of the lattice are produced. These are given in figure 3.1: (12345, Ø), (1256, a), (1346, e), (12, ac), (16, ae), (346, be), (1, acde), (6, abe), (Ø, abcde). Algorithm 3 determines where to add edges between concepts. For example, we let $F = \{1, 2, 5, 6\}$. So we first set $\text{COUNT}(F)$ to 0. Then for the first element in $B$—$a' = \{1, 2, 5, 6\}$, $F' = F \cup m' = \{1, 2, 5, 6\}$, set $\text{COUNT}(\{1, 2, 5, 6\})$ to 1, and check whether the number of elements in $\gamma(F')$ equals $\text{COUNT}(F') + \text{the number of elements in } \gamma(F)$. Here, $\gamma(F') = \gamma(F) = \text{COUNT}(F') = 1$, so the result is false, and we reset $\text{COUNT}$ to 0. For the second $m'$—$b' = \{3, 4, 6\}$, $F' = \{6\}$, $|\gamma(F')| = 3 \neq |\gamma(F)| + 1 = 2$. For the third $m'$—$c' = \{1, 2\}$, $F' = \{1, 2\}$, $|\gamma(F')| = 2 = |\gamma(F)| + 1$, so we add an edge from (1256, a) to (12, ac). For the fourth $m'$—$d' = \{1\}$, $F' = \{1\}$, $|\gamma(F')| = 4 \neq |\gamma(F)| + 1 = 2$. For the fifth $m'$—$e' = (1, 3, 4, 6)$, $F' = \{1, 6\}$, $|\gamma(F')| = 2 = |\gamma(F)| + 1$, so we add an edge from (1256, a) to (16, ae).

The algorithm to determine the formal concepts provided in (Troy et al. 2007) follows basically the same approach as that in (Qiao et al. 2003) but its orientation is from the object perspective and only maintains the intent (or the extent) of the formal concept throughout the algorithm. The first stage, stage 0, in this algorithm parallels Qiao’s Algorithm 1 in that a “basis” is created except that this basis computes all attributes for each object instead of all objects for each attribute. All concepts in the basis are inserted into a concept set $C$. Then in each following stage, it always computes the intersection of each pair of concepts in the previous stage in order to find out any new set of attributes and put these new concepts into the concept set $C$. When there is only one concept in the current stage, the concept family is generated, and it equals the concept set $C$.

\[
\text{begin}
\]

\[
i insert M in SearchTree
\]

\[
\text{end}
\]

\[
\text{reset COUNT}
\]

\[
\text{end}
\]
insert each member of \( C_i \) in SearchTree if it is not already in,

where \( C_i := \{ \{ g \} | g \in G \} \)

\( i = 2 \)

while \( |C_{i-1}| > 1 \) do

begin

\( C_i = \{} \)

for each pair of (distinct) concepts \( c_j, c_k \) in \( C_{i-1} \) do

begin

\( \text{Candidate} = c_j \cap c_k \)

if candidate not in SearchTree then \( C_i = C_i \cup \{ \text{candidate} \} \)

\( i++ \)

end

end

\( C = \cap_i C_i \)

End

Using the same example in figure 3.1, \( C_0 = \{ a, b, c, d, e \} \), \( C_1 = \{ \{ a, c, d, e \}, \{ a, c \}, \{ b, e \}, \{ a \}, \{ a, b, e \} \} \), \( C_2 = \{ \{ a, e \}, \emptyset, \{ e \} \} \), and \( C_3 = \emptyset \) where \( |C_3| = 0 \). The loop ends, and the final \( C = C_0 + C_1 + C_2 + C_3 = \{ \{ a, b, c, d, e \}, \{ a, c, d, e \}, \{ a, c \}, \{ b, e \}, \{ a \}, \{ a, b, e \}, \{ a, e \}, \emptyset, \{ e \} \} \) which is the set of intents of all formal concepts in this lattice. However, the presented algorithm does not include a way to calculate the extent for each of the formal concepts or a method to construct the concept lattice. It uses Pritchard’s algorithm (Pritchard 1999) for constructing the concept lattice with intents only.

Comparing the above algorithm with Qiao’s algorithm 2, one sees that there are no significant differences between these two. The only difference is that Qiao uses all related objects of each attribute as the basis \( B \), while Troy uses all related attributes of each object as \( C_1 \). This serves as
the basis B. The algorithms in Qiao’s paper also provide the means to construct the complete formal concept, i.e., both its intent and extent and the structure of the concept lattice.

4.3 Combining Rough Set Theory and Concept Lattices

In (Yao 2004) rough set approximation operators are defined with respect to a formal context. Using these approximation operators, three variations of the general concept lattice built from a formal context are presented: the object-oriented concept lattice (OOCL), the property-oriented concept lattice (POCL), and the complement of the formal context concept lattice (CFCCL). The relationships between these three lattices and the standard concept lattice are explained.

Rough set theory is introduced through the proposal of approximation operators for formal concept analysis. In a formal context (U, V, R), the binary relation can be defined in these two forms (U corresponds to G, V to M, and R to I in the definitions presented in section 3):

\[ xR = \{ y \in V \mid xRy \subseteq V \} \]
\[ Ry = \{ x \in U \mid xRy \subseteq U \} \]

The set xR can be extended to a subset \( X \subseteq U \) and Ry to a subset \( Y \subseteq V \):

\[ XR = \bigcup_{x \in X} xR \]
\[ RY = \bigcup_{y \in Y} Ry \]

With respect to a formal context (U, V, R), a pair of dual approximation operators from \( 2^U \) to \( 2^V \) are defined as follows:

\[ X^\square = \{ y \in V \mid \forall x \in U (xRy \rightarrow x \in X) \} \]
\[ = \{ y \in V \mid Ry \subseteq X \} \]
\[X^\phi = \{y \in V \mid x \in U (xRy \land x \in X)\}
= \{y \in V \mid Ry \cup X \neq \emptyset\}
= \bigcup_{x \in X} xR
= XR\]

Conversely, we can also define a pair of dual approximation operators from \(2^V\) to \(2^U\), that is \(Y^\Box\) and \(Y^\phi\).

Based on the notion of approximation operators, the object oriented formal concept is introduced as \(X = Y^\phi\) and \(Y = X^\Box\). \(X\) is the extent of the concept and \(Y\) is the intent. And similarly, a property oriented formal concept is introduced as \(X = Y^\Box\) and \(Y = X^\phi\). The corresponding concept lattices the OOCL and POCL can be created. The two new concept lattices can be integrated with the general concept lattice, framework as follows: a formal concept can be defined as \(X = X^\phi \Box\) and \(Y = Y^\Box \phi\).

The use of approximation operators provides a disjunctive description of the knowledge embedded within a formal context where the formal concept analysis approach provides a conjunctive description of that knowledge. The conclusion of the paper suggests that a combination of the two approaches might provide more insight into understanding the data. An objective of this thesis research is to explore the use of these other two approaches OOCL and POCL to provide other views of the data and explore their relationships.

In (Zhao et al., 2006) FCA and rough set theory are combined in order to develop a new similarity measure that could be used in the task of mapping between ontologies. The design objective of this similarity model is to improve both the mapping accuracy by incorporating both the structural and feature information gained from FCA and the similarity performance using rough
set theory. In this approach one formal context is created where the objects are the terms from a thesaurus and the attributes are other terms used in defining or describing these terms.

In (Zhao et al. 2006), it states “For a subset of objects \( C \subseteq U \), its lower attribute approximations are defined by:” (Note: \( U \) represents the set of objects for the formal context which is also labeled \( G \) in the description of concept lattices in Chapter 3.)

\[
C_{LA} = \text{extent} \left( \bigvee \{(X, Y) \in L \mid X \subseteq C\} \right)
\]

This statement is incorrect since taking the extent of a formal concept results in a set of objects, not in a set of attributes. Basically, what the above formula provides is the lower object approximation for a set of objects from the universe of objects. This value is approximated by the extent of the join of all formal concepts whose extents are subsets of the set \( C \). For example, in the concept lattice in figure 3.1, let \( C = \{1, 2, 4, 5, 6\} \), then \( \{(X, Y) \in L \mid X \subseteq C\} = \{(1256, a), (12, ac), (16, ae), (1, acde), (6, abe), (\emptyset, abcde)\} \). The join of all these concepts is (1256, a). So \( C_{LA} = \{1, 2, 5, 6\} \).

Based on this lower object approximation, the object-based similarity measure model is described as follows:

\[
sim(a, b) = \frac{|(a \lor b)_{LA}|}{|\text{(a \lor b)}_{LA}| + a|a_{LA} - b_{LA}| + (1 - a)|b_{LA} - a_{LA}|}
\]

When we compute the \((a \lor b)_{LA}\), the join of concepts \( a \) and \( b \) is performed and then the extent of this join is used as the set of objects \( C \) in the above formula for \( C_{LA} \). Note that this formula is given in (Zhao et al. 2006) but they refer to it as attribute-based similarity measure. The discussion in (Zhao et al. 2006) is emphasizing attribute similarity and the examples they provide work with the intents or attributes of the concepts. Their discussion is confusing because in one place they refer to \( C \) as a set of objects and then in another place they refer to \( C \) as a set of concepts. However, attribute based similarity can be measured. The lower approximation for a
set of attributes $D \subseteq M$, is given using the formula $D_{LA} = \text{intent } (\bigwedge \{ (X, Y) \in L \mid D \subseteq Y \})$ (Yao et al. 2004) in place of the formula for $C_{LA}$.

### 4.4 Applying Concept Lattice Representations in Bioinformatics

Recently researchers are using concept lattice representations as an effective tool for analyzing large and high dimensional biomedical and bioinformatics data. This section summarizes two papers, one specific to the creation of concept lattices from annotated taxonomies with an example using the GO (Kaiser et al. 2006) and the other which presents an algorithm to analyze large biomedical data (Fu et al. 2007).

In (Kaiser et al. 2006) an annotated ordered set is defined and how to use it in modeling large taxonomically structured ontologies such as the Gene Ontology is demonstrated. Previously in Figure 2.2 a sample portion of the GO with annotated genes is illustrated. Nodes in black represent functional categories of biological processes, basically things that proteins “do”. Nodes are connected by links indicating subsumptive, “is-a” relations between categories, so that, for example, “DNA ligation” is a kind of “DNA repair”. GO, nodes can also be connected by compositional, “has-part” relations, but in most research the GO links are considered to be the same type. If one consider biological process terms as attributes in a formal context and considers genes in colors as the objects, and its corresponding concept lattice can be created as follows:
The authors propose that an application of this research is to consider the how formal concept analysis on the actual annotation usage of a controlled vocabulary in the form of an ontology might be used to review or refine the ontology itself. Future research possibilities include developing measures and tools to assess a degree of conceptual soundness and completeness based on a comparison of the concept lattice to the original ontology whose terms are used in the annotated objects in creating the concept lattice. The objective of this thesis research is somewhat different. FCA is to be used as another approach to determine relationships and similarities between genes and gene products and between various sets of genes or gene products.
In (Fu et al. 2007), the use of FCA for data analysis of biomedical data sets is promoted and a lattice-based algorithm for such analysis is presented. The authors emphasize the feature of duality of concept lattices because the performance of a concept-lattice building algorithm is dependent on the number of objects and attributes. They simply propose that the relation \( I \) between objects and attributes be inverted by switching the rows and columns. They point out that this is very useful for example with gene expression datasets where there may be 1000 to 10000 attributes but only 100 to 1000 objects.

The one author H. Fu had previously (Fu et al. 2003) proposed a scalable lattice-based algorithm ScalingNextClosure to decompose the search space for finding formal concepts in large datasets into partitions and then generate independently concepts (or closed item sets) in each partition. The experimental results reported show that ScalingNextClosure algorithm is scalable for large datasets. Their algorithm only shortens the execution time of determining all formal concepts, but no improvement is made in the execution time of the construction of lattice (adding edges between concepts). The actual concept lattice construction is the more time-consuming task for the whole procedure so that more research efforts are needed to investigate parallel algorithms to build the concept lattice.

### 4.5 Concept Approximation Theory

The process of FCA produces a set of formal concepts and the relationships between these formal concepts in the structure of the concept lattice. In some real world situations, an entity might exist described both by a set of objects and a set of attributes but that does not actually map to a formal concept existing in a concept lattice. Such an entity is referred to as a non-definable concept (Saquer and Deogun, 2001). Concept approximation is the process of finding formal concepts that best describe non-definable concepts. This section introduces several different ways to calculate the lower approximation (LA) and upper approximation (UA) of a
concept that may or may not exist in the concept lattice. Section 5.4 provides examples using these various methods based on their implementation of the concept lattice querying features and compares the results.

4.5.1 Rough Set Theory Concept Approximation

In (Saquer and Deogun, 2001), rough set theory is used to develop an approach to concept approximation. The objective is to determine the formal concepts that approximate a single set of objects, a single set of attributes or a pair of sets, one of objects and one of attributes. For a set of object A, the lower approximation (LA) of A is given as

$$\text{LA}_\text{RSQ}(A) = \bigcup \{X \in G/R \mid X \subseteq A\},$$

and the upper approximation of A (UA) is given as

$$\text{UA}_\text{RSQ}(A) = \bigcup \{X \in G/R \mid X \cap A \neq \emptyset\},$$

where G/R refers to the equivalence classes of the formal context where R is defined between two objects g_1 and g_2 as g_1Rg_2 iff g_1I = g_2I where gI = \{m \in M | gIm\}. G/R is the set of all equivalence classes induced by R on G.

The lower and upper approximations for set of objects A do not need to be calculated if the set A is the extent of a formal concept, i.e., (A, A') exists as a concept in the concept lattice. The best concept approximation for A is (A, A'). If LA(A) = UA(A), then the concept approximation for the set of objects A is simply (\[\text{UA}(A)\]', [\text{UA}(A)]'). Otherwise the lower concept approximation for the set A of objects is given by (\[\text{LA}(A)\]', [\text{LA}(A)]') and its upper concept approximation is given by (\[\text{UA}(A)\]', [\text{UA}(A)]').

Similarly in (Saquer and Deogun, 2001), a lower and upper approximation can be determined for a set of attributes B. In the above equations for lower and upper approximation for a set of objects A, substitute B for A, substitute m for g and replace R with R' such that R' is defined between two attributes m_1 and m_2 as m_1R'm_2 iff Im_1 = Im_2 where Im = \{g \in G | gIm\}. M/R' is the set of all equivalence classes induced by R' on M. The one difference is that the
lower concept approximation for a set of attributes B is given by ([UA(B)]', [[UA(B)]]') and the upper concept approximation is given by (LA(B)', [[LA(B)]]'). This modification can be explained by the structure of the concept lattice where the size of the extents decrease in size as one descends the concept lattice whereas the size of the intents increase in size as one descends the concept lattice.

Given a pair (A, B), where A is a set of objects and B is a set of attributes, formal concept(s) can also be approximated for this pair. There are four cases to consider depending on whether A and B represent the extent and intent of any of the formal concepts. The details of this algorithm can be found in (Saquer and Deogun, 2001).

### 4.5.2 Concept Approximation using Irreducibility

The idea of irreducibility is introduced into the formulation of concept approximation in Hu’s approach (Hu et al., 2001). The relation R used in Saquer’s LA and UA operations is modified to S such that g1Sg2 iff g1I ⊆ g2I. The relation S is not an equivalence relation on G but instead a partial relation on G. It is shown that every pair ([g]S, ([g]S)') where [g]S = {g ∈ G | gSg} is a join-irreducible concept in the concept lattice. Let JI be the set of all join-irreducible concepts, then JI is equivalent to the set of all pairs ([g]S, ([g]S)').

MI, the set of all meet-irreducible concepts is shown to be equivalent to the set of all pairs (([m]S)', [m]S)') with the relation S' defined between two attributes m1 and m2 as m1S'm2 iff Im1 ⊆ Im2 where Im = {g ∈ G | gIm}. Then given a set of objects A, the best lower concept and upper concept approximations are given by:

\[
LA_{Hu}(A) = \bigcup\{X \subseteq A \mid (X, X') \in JI\},
\]

\[
UA_{Hu}(A) = \bigcap\{A \subseteq X \mid (X, X') \in MI\}
\]

JI and MI each represents join-irreducible and meet-irreducible nodes in concept lattice. The lower approximation concept and upper approximation concept that best describe set A are given by (LA(A), (LA(A))') and (UA(A), (UA(A))') respectively.
Similarly a lower and upper approximation for a set of attributes $B$ is defined as

$$LA_{Hu}(B) = \bigcap \{ B \subseteq Y \mid (Y', Y) \in MI \}$$

$$UA_{Hu}(B) = \bigcup \{ Y \subseteq B \mid (Y', Y) \in JI \}$$

Then given a set of objects $B$, the best lower concept and upper concept approximations are given by $((LA(B))', LA(B))$ and $((UA(B))', UA(B))$, respectively. In (Hu et al., 2001) no lower and upper approximation concepts for any arbitrary pair $(A, B)$ are provided since these researchers state that it might not represent any meaningful concept, for example the pair $(G, M)$ is meaningless.

### 4.5.3 Concept Approximation using Lattice-Theoretic Operators

In (Yao and Chen, 2004), the objective is to approximate a set of objects $A$ by the extensions of a pair of formal concepts from the concept lattice. The lower concept approximation is to be the largest concept whose extension is a subset of $A$ and that the upper concept approximation is to be the smallest concept whose extension is a superset of $A$. In order to satisfy this requirement the lower and upper approximations for $A$ are derived using the lattice operators of join and meet respectively and given as:

$$LA_{Yao}(A) = \text{extent}(\bigvee \{(X, Y) \in L \mid X \subseteq A\})$$

$$UA_{Yao}(A) = \text{extent}(\bigwedge \{(X, Y) \in L \mid A \subseteq X\})$$

where $L$ consists of the set of formal concepts making up the concept lattice. Then the lower concept approximation is defined as $(LA_{Yao}(A), (LA_{Yao}(A))')$ and the upper concept approximation as $(UA_{Yao}(A), (UA_{Yao}(A))')$.

In comparison of Yao’s method to Hu’s approach, Hu’s method only uses formal concepts in the concept lattice that are join-irreducible for the lower approximation of $A$ and those that are meet-irreducible for the upper approximation of $A$. In Yao’s method, all formal concepts in the concept lattice are selected. Hu’s method uses set-theoretic operators instead of the lattice-theoretic operators used in Yao’s approach. Although lattice-theoretic operators of meet and
join are defined based on the set-theoretic operators of intersection and union, they are not the same. An intersection of extensions (intensions) of a set of concepts is always the extension (intension) of a concept. But a union of extensions (intensions) of a set of concepts is not necessarily the extension (intension) of a concept.

As part of this thesis research, an investigation of these various concept approximation methods has been performed.

4.6 Similarity Measures between Concepts in a Concept Lattice

The fourth query in QUOTA takes as input two objects each described by a set of annotated objects and produces the similarity between the two input objects. This similarity is based on the similarity between the two sets of annotated objects. Another approach is to create a separate concept lattice for each set of annotated objects and then assess the similarity between the two objects based on assessing the similarity between their respective concept lattices. To apply this approach, similarity between the individual concepts in one concept lattice to those in another must be measured. This section discusses various methods of similarity assessment between formal concepts. It first introduces the model which serves as the foundation for many concept similarity measures.

4.6.1 A Set-theoretic Basis

Although numerous methods for measuring similarity between two different objects which are represented as a set of features have been proposed throughout the years, the basic model that generalizes these measures is Tversky’s parameterized ratio model of similarity [2] given as

\[
S_r(X,Y) = \frac{f(X \cap Y)}{f(X \cap Y) + \alpha * f(X - Y) + \beta * f(Y - X)}
\]
X and Y are sets of the respective features of objects x and y. The function f is an additive function on disjoint sets, i.e., set cardinality. This measure is normalized so that $0 \leq S(X, Y) \leq 1$. With $\alpha=\beta=1$, S becomes the Jaccard index which similarity measure that is symmetric, i.e., neither object is considered the reference object.

$$S_{\text{jaccard}} (X, Y) = \frac{f(X \cap Y)}{f(X \cup Y)}$$

With $\alpha=1$, $\beta=0$, S becomes the degree of inclusion for X, the proportion of X overlapping with Y. It is given as

$$S_{\text{inclusion}} (X, Y) = \frac{f(X \cap Y)}{f(X)}$$

In this case, the reference object is Y since it is determining how much of X is described by Y. Similarly with $\alpha=0$, $\beta=1$, $S_{\text{inclusion}}$ becomes the degree of inclusion for Y, the proportion of Y overlapping with X. Clearly, the inclusion measure is not symmetric.

### 4.6.2 Concept Similarity

Concepts are described by their two components, their intents and their extents. It is natural to assess the similarity between two concepts based on an aggregation of their intent similarity and their extent similarity. Since both of these are sets, the Tversky ratio model can easily be used in creating measures for assessing concept similarity. Below is a weighted aggregation of the Jaccard measure of the concept similarity between concepts $X$ and $Y$:

$$S_{\text{C-JacWeight}} (X, Y) = \omega \frac{|E(X) \cap E(Y)|}{|E(X) \cup E(Y)|} + (1 - \omega) \frac{|I(X) \cap I(Y)|}{|I(X) \cup I(Y)|}$$

where $E(X)$ represents the extent set of concept $X$ and $I(X)$ represents the intent of concept $X$. By using the $\omega$ parameter, one could determine concept similarity based solely on the extents ($\omega=1$) or solely on the intents ($\omega=0$). One could also easily replace the Jaccard similarity measure with an inclusion measure if one of the concepts is considered the referenced concept.

In (Saquer and Deogun, 2001) Jacard’s measure is also used but the focus is on the use of similarity measures to determine which concept in a concept lattice best approximates a set of objects A and a set of attributes B. The objective of the above aggregated model of concept
similarity is to simply measure similarity between two concepts already existing in a concept lattice. The set A of objects and the set B of attributes do not necessarily exist in the concept lattice as an extent or an intent of an existing concept.

Measures of concept similarity that are variation on the above equation for $S_{C-JacWeight}$ using different weighting schemes or different similarity measures between sets do not incorporate the structure of the concept lattice but only considers the internal make up, i.e., the extent and intent of the formal concepts. Other concept similarity measures have incorporated some structure information of the concept lattice in the similarity assessment by considering two different factors such as irreducibility (Souza and Davis, 2004), lower approximations (Zhao et al., 2006) and both (Wang and Liu, 2008). A discussion of these approaches is provided in (Cross and Yi, 2009).

In this thesis research, the above set-theoretic approach serves as the basis for assessing concept similarity. One additional consideration, however, is that for annotated objects, the set of attributes are taken from an ontology such as the Gene Ontology. In (Formica 2008), the measure $S_{C-JacWeight}$ is modified to use inclusion indices instead and the method used in the inclusion index on the intents of the concepts is changed to incorporate the semantic similarity of the attributes between the two sets. The following equation defines this concept similarity measure:

$$S_{C-JacSS}(X, Y) = \omega \frac{|E(X) \cap E(Y)|}{r} + (1 - \omega) \frac{M(I(X), I(Y))}{m}$$

In this formula, the first component is like $S_{C-JacWeight}$ except that the normalization factor is $r$ instead of $|E(X) \cup E(Y)|$ in the Jaccard index. This modification causes this component to be an inclusion measure since $r$ is selected as max($E(X), E(Y)$). Similarly for the intent component of the above measure $m$ is max($I(X), I(Y)$). The difference is that the numerator for the intent component is not simply $|I(X) \cap I(Y)|$ as in $S_{C-JacWeight}$. Instead it uses $M(I(X), I(Y))$ which is defined as the maximum sum of information content (IC) semantic similarity of all possible sets of pairs of attributes that can be formed between the two intent sets where the first one is taken from $I(X)$ and the second one is taken from $I(Y)$. Using semantic similarity between pairs of attributes is an existing approach to assess similarity between genes or gene products and
is incorporated in QUOTA (Cross and Sun, 2007). The only difference in (Formica, 2008) is a different method is used to aggregate the semantic similarities for the pairs of attributes.

As explained in (Cross and Sun, 2007) two standard ways for aggregating these semantic similarities is the average of all pairs and the average of the maximums of the pairs. For example, let $I(X) = \{x_1, x_2, x_3\}$ and $I(Y) = \{y_1, y_2, y_3, y_4, y_5\}$. In the following two equations, $nX$ is $|I(X)|$, $nY$ is $|I(Y)|$, and $SS(x_i, y_j)$ is an assessment of the semantic similarity between the two attributes. $SS(x_i, y_j)$ assumes that an ontology exists that contains the attributes used in the intents of the concepts and can be used in determining the semantic similarity between the attributes. The average of all pairwise similarities is expressed as:

$$S_{PW-ave}(X, Y) = \frac{\sum_{i=1}^{nX} \sum_{j=1}^{nY} SS(x_i, y_j)}{nX \times nY}$$

The average of the maximum pairwise similarities is expressed as

$$S_{PW-max-ave}(X, Y) = \frac{\sum_{i=1}^{nX} \max_{j=1}^{nY} SS(x_i, y_j) + \sum_{j=1}^{nY} \max_{i=1}^{nX} SS(x_i, y_j)}{nX + nY}$$

In (Formica, 2008), they instead propose another approach to aggregation based on the maximum weighted matching problem (Galil 1986) which is represented by their function $M(I(X), I(Y))$. The objective is to maximize the sum of the semantic similarity measures between pairs of attributes, one from first set and the other from the second set with the constraint that all attribute pairings have a unique first attribute and a unique second attribute. Then the maximal sum of these semantic similarities is averaged.

### 4.7 Assessing Concept Lattice Similarity

Recently, research in bioinformatics has used an algorithm to assess the distance between two concept lattices (Choi et al. 2007) for use in microarray data analysis. The microarray data is expressed in a concept lattice with each node in the concept lattice corresponding to a subset of genes grouped together based on expression values and other biological information related to gene function. The similarities and differences between two concept lattices created from
microarray experiments can be used in investigating the similarities and differences in the experiments.

Their approach taken to assess the difference between two concept lattices uses a simple distance based on common subgraphs. In this problem domain the concept in the concept lattice is made up with its extent being a subset of genes and its intent being subset of attributes; however, only the extent is considered as the label for each vertex, i.e., concept in the concept lattice. Assume two concept lattices $L_1 = (V_1, E_1)$ and $L_2 = (V_2, E_2)$, where $V_1$ is the set of vertices, i.e., concepts, in $L_1$ and $E_1$ is the set of edges in $L_1$. A vertex $v$ is considered a common vertex if it appears in both lattices. Let $V_C$ be the set of common vertices, that is $V_C = V_1 \cap V_2$. If vertices $u$ and $v$ are in $V_C$ and the edge $e = (u,v)$ is in both $E_1$ and $E_2$, then $e$ is considered a common edge. The set of common edges is $E_C$. The distance between $L_1$ and $L_2$ is then defined as

$$
\frac{|L_1 \setminus L_2| + |L_2 \setminus L_1|}{|L_1 \cup L_2|}
$$

where

$$
|L_1 \setminus L_2| = |V_1 \setminus V_C| + |E_1 \setminus E_C|
$$

$$
|L_2 \setminus L_1| = |V_2 \setminus V_C| + |E_2 \setminus E_C|
$$

$$
|L_1 \cup L_2| = |L_1| + |L_2 \setminus L_1| \text{ with } |L_1| = |V_1| + |E_1|
$$

Because it was not possible to find perfect matches on vertex labels, i.e. one concept’s extent in $L_1$ matching exactly to another concept’s extent in $L_2$, the researchers considered two vertices, $v_1$ and $v_2$ the same if $|\text{extent}(v_1) \cap \text{extent}(v_2)| \geq \xi \max(\text{extent}(v_1), \text{extent}(v_2))$, that is their inclusion index is greater than $\xi$.

In the conclusion of the paper, they state that their results show using FCA approach for comparing microarray data experiments is promising but that their distance measure is basic and does not incorporate the structure of the concept lattices.

In (Joslyn et al. 2008) a method to measure the structural quality of alignments between semantic hierarchies is presented. Their objective is to evaluate a proposed mapping between
concepts from one ontology $O_1$ to concepts in another ontology $O_2$. They evaluate the structural quality of the mapping using two criteria: order preservation and distance preservation. With order preservation, the mapping should not alter the hierarchical ordering between concepts, i.e., if one concept appears above another concept in $O_1$ then the concepts that these two are mapped to in $O_2$ should have that same ordering. With distance preservation, the distance between two concepts in $O_1$ should be close to the distance between the two concepts they are mapped to in $O_2$. These same ideas are applied into determining the similarity between two concept lattices in this FCA software. The details are presented in section 5.5.

5 Concept Lattice Creation and Querying Subsystem of QUOTA

The main objective of this thesis research is the development of a software tab that is incorporated into the QUOTA system that can be used to perform formal concept analysis on a formal context developed from a set of annotated objects where the annotations are taken from an ontological terminology. This software has been developed to provide multiple views of a formal context and to be able to perform a variety of queries on the resulting concept lattices. It incorporates methods of assessing similarity between individual formal concepts and between complete concept lattices.

Based on a survey of the research literature, except for the algorithm for building general concept lattices, these algorithms have not been implemented in a software system available for general use or in the context of annotated objects. In addition, some descriptions in the research literature were often unclear, simply mathematical definitions and provided no clear algorithm on how to calculate the values. This section explains in more detail the implementation of these algorithms and the changes made to incorporate them into the QUOTA system. The algorithm to construct granulated concept lattices from an existing concept lattice with its associated ontological terminology is a result of this thesis research. The development of the relationships between the various concept approximation methods and their use in concept similarity and
concept lattice similarity assessment is also a result of this thesis research. Those algorithms which have already been more fully described in the previous related research sections are only briefly reviewed here for completeness.

5.1 Concept Lattice Creation

As previously discussed there are numerous algorithms for building concept lattices. As part of this thesis research, these algorithms were studied and the two algorithms described in section 4.2 have been implemented in Java. A comparison of the complexities of these two algorithms revealed that which of the two is faster depends on whether the number of objects or the number of attributes in the formal context is larger. This result also agrees the dual algorithm principle which states that a dual algorithm of an algorithm \( A \) is the application of \( A \) to the transposed context (Fu et al. 2007). The implemented Java algorithm first determines the numbers of objects and attributes in the formal context in order to select the appropriate algorithm. If the number of objects is larger than the number of attributes, Qiao’s algorithm is used to calculate the formal concepts and finally create the concept lattice; otherwise Troy’s algorithm is used to determine the basis which is a set of sets containing attributes instead of objects as in Qiao’s algorithm.

No matter which algorithm is used, the main steps are basically the same and only differ on what basis is calculated: 1) calculate the basis of the formal context, 2) generate the family, 3) calculate the extent and the intent of each concept, 4) compare each concepts to find “super” and “sub” relations, and 5) finally create edges among concepts.

These algorithms require that a formal context be input but the formal context can be created for any domain. Since this software is integrated with the QUOTA system, the natural domain being used to create a formal context is an annotation file containing a set of objects, each being described by the annotation terms mapped to it. For example, using with the GO analysis, the objects represent gene or proteins and the attributes are the GO terms found within an annotation file(s). The formal context is created from an annotation file that maps GO terms
to gene or gene products. Additional program code may be used if names instead of unique numeric identification are used to identify the genes or gene products. For example, the contents of GeneName2ID.txt file (www.geneontology.com) is used to map a gene name to gene numeric ID. A list of unique gene IDs is determined from the annotation file. Then two hash tables are created. The first one maps each gene ID to its corresponding annotating GO term IDs; the second one maps each GO ID to its corresponding gene IDs. Using these hash tables, the basis, family, edges and concept lattice are created using the concept lattice building algorithms.

The design of the software to build the concept lattices initially used a two-dimension array to store a formal context table, but this greatly limited the size for and processing speed on the formal context. The design was changed to use two hash tables, one to store all corresponding attributes for each object and the other to store all corresponding objects for each attribute. This design uses much less space and is much faster for calculating the basis and many other processing steps in building the concept lattice.

Besides the formal context, the concept lattice is stored in hash tables as well. It maps each single concept to its child concepts so that the hierarchy and the parent-child relationship between two concepts in the concept lattice are easily represented. Hash tables are especially useful when dealing with huge annotation files. It further shortens the running time after the complexity of the algorithm is already simplified.

The algorithms to create multiple views of the same formal context are based on the definition of the object-oriented concept lattice (OOCL), the property oriented concept lattice (POCL), and the Complement concept lattice (CCL) presented in (Yao 2004) which only provided definitions of operators and examples to explain these new kinds of concept lattices. No algorithms were provided to create the OOCL or the POCL. Creating the complement concept lattice of a formal context is fairly easy and simply requires reversing the entries in the matrix representing the original formal context, i.e where there is an association between an object and attribute, remove the association and add an association where there is not an association. The
complement concept lattice can then be built from the reversed formal context by using the
general concept lattice building algorithm.

To determine an algorithm for POCL and OOCL, the figures given for examples in (Yao
2004) were examined carefully. This examination showed that the extent of each concept in the
CCL matched the extent of a concept in the POCL and the intent of each concept in the CCL
matched the intent of a concept in the OOCL. Therefore, the key to creating the POCL and the
OOCL appeared to be the creation of the complement concept lattice. Because this was only
based on looking at a few examples, a rationale for why this was true and if it was true in general
needed to be determined. This relationship of the CCL to the POCL and its relationship to the
OOCL were determined to be true in general and explainable.

The CCL produces formal concepts, i.e., (X, Y) such that all objects in the extent X of a formal
concept do not possess any of the attributes in its intent Y. The extent X of the corresponding
formal concept in the POCL are exactly those objects that possess a subset of all attributes
except for those in Y, i.e., M - Y. Similarly, the intent Y the corresponding formal concept in
the OOCL are exactly those attributes that are possessed by a subset of all objects except for
those in X, i.e., G - X. To summarize, the CCL can be used to easily build the OOCL and
POCL of the formal context used to build the original concept lattice. The CCL, the OOCL, and
the POCL all have same numbers of concepts; the OOCL shares the same intents of all concepts
in the CCL, and the POCL shares the same extents of all concepts in the CCL.

Since the basic concept lattice algorithms are implemented, the procedure to produce the POCL
is to first take the complement of the formal context and form the CCL. This step produces the
correct structure for the POCL and the correct extents for its formal concepts. Now assume a
concept in the POCL, say (A, B)o corresponds to (X, Y)c where the subscripts p and c indicate in
the POCL and in the CCL, respectively. It is known that A=X. The intent B can be determined
as \( B = A^\Diamond \) by definition of POCL or also more simply by \( B = Y^c \), the complement of intent for the corresponding concept in the complement concept lattice.

The way to produce OOCL is very similar with POCL. The only difference is that it is the intent, not extent, of each concept in OOCL being exactly the same as the intent of corresponding concept in the CCL, i.e., a concept in the OOCL, say \((A, B)_o\), corresponds to \((X, Y)_c\) and \( B = Y \). The extent \( A \) can be determined as \( A = B^\Diamond \) by definition of OOCL or also more simply by \( A = X^c \).

### 5.2 Filtered and Granulated Concept Lattice Construction

In a formal context, the properties are considered to be independent. The algorithms building the previously described concept lattices make this assumption. No additional information from the structure of the ontological terminology is incorporated. An object is only associated specifically with the terms that are directly used to annotate it and the relationships between the terms that exist within the ontological terminology are ignored.

For example, from Figure 2-2 if using the standard assumption of attributes as independent from one another, gene object Recc1 is annotated by the GO term, “DNA strand elongation” and gene object CDC9 is annotated by term “DNA litigation” and they do not share a common property, i.e., the structure of the ontology is ignored. Another scenario is that the gene objects Recc and CDC2 are annotated by terms “DNA strand elongation” and “leading strand elongation” respectively. If just their direct annotations are used, these two gene objects do not share a common property, but “DNA strand elongation” is actually an ancestor (specifically parent) of “leading strand elongation.” In the standard approach which uses only direct annotations to build the concept lattice, the ontology structure has been ignored.

This thesis research has developed an algorithm to use the structure of the ontology to create granulated concept lattices from the original concept lattice without rebuilding the concept lattice.
from a new formal context. The following sections describe this new concept lattice modification method that uses an associated ontological terminology.

5.2.1 Constraining the Selection of Annotations

The formal concept analysis tool using annotations as objects’ attributes where the annotations come from an ontological terminology allows several different approaches that incorporate the ontology’s structure in the creation of the formal context. There are two different features provided to construct the concept lattice using the ontological terminology. The first is filtering out direct annotations from the constructed concept lattice based on a selection criterion. For filtering, the original concept lattice is modified to remove the attributes, i.e., annotations that do not meet the selection criteria. The second is granulating direct annotation from the constructed concept lattice based on the selection criteria. Granulating does not totally remove direct annotations as in the filtering techniques but instead replaces them with their most immediate ancestor term that meets the selection criteria. For both cases, however, a method of specifying the selection criteria is needed.

For granulation, the selection criterion is needed to limit the number of ancestors replacing the direct annotations. Including all the ancestors of each direct annotation greatly increases the number of properties of the formal context. Several different methods of limiting the inclusion of ancestor terms are possible. For example, one might limit by a specific number how many ancestor generations to go back for term inclusion. Another approach might limit the number of ancestors to use for term inclusion by the information content of the ancestor term within the ontology. Another aspect of using the structure of the ontology is to filter even the direct annotations based on characteristics of the annotating term. For example, given a set of gene objects and their annotating terms, instead of creating the a full concept lattice, the concept lattice might be created by only using annotating terms as properties for the formal context if
they pass filtering parameters such as the information content (IC), the level, or interval rank (IR) of a term. These three node metrics for an ontology exist in QUOTA system.

Level is simply the depth at which the term occurs in the ontology. The information content for a term in a hierarchy is given as:

\[
IC(n) = \log \frac{[\text{hypo}(n) + 1]/\text{max}] \log (1/\max)}{\log (\max)} = 1 - \log \frac{\text{hypo}(n) + 1}{\log (\max)}
\]

where hypo(n) is the number of descendents of term n, and max is the maximum number of terms in the hierarchy. The IC value of a leaf term is 1 because it has no descendants. As one proceeds up the hierarchy to the root term, the IC value decreases gradually to 0. The interval rank value is given as

\[
IR(n) = (\text{depth}(n) + M - \text{height}(n))/2
\]

where M is the maximum path length from root term to a leaf term.

The new granulation feature provided as part of this thesis research relies on these existing metrics for filtering the direct annotations selected for creating the original concept lattice. In addition these metrics are used for incorporating ancestor terms that satisfy the criteria when their descendant terms that are direct annotations of an object do not satisfy the criteria. By filtering or granulating the concept lattice, the user can see how the concept lattice changes based on modifying the selection criteria and might be able to identify certain relationships between genes more readily based on the change in the concept lattice.

When forming a granulated concept lattice for an already existing full concept lattice, the standard way would be to recreate the formal context using the modified set of annotations resulting from applying the selection criteria. Then using this new formal context, a new concept lattice can be constructed from scratch. The disadvantage of this approach is that it can be time-consuming to rebuild an entire new concept lattice. As part of this thesis, another approach is developed that instead modifies the origin concept lattice to a granulated one based on the substitution of the satisfying ancestor term(s) for the unsatisfying annotation attributes in the
intents of the concepts of the original CL. This new approach to creating concept lattice from an existing concept lattice is the first one that takes into consideration the complete structure of the ontological terminology and does not just consider annotations that are leaf concepts in an ontological terminology (Hashemi et al. 2004).

5.2.2 Granulated Concept Lattice Algorithm

Before looking at the granulated concept lattice algorithm, several data structure used often in the algorithm need to be introduced:

**Property (attribute) hash table:** the key is the attribute identifier, and the entry for each attribute is the list of concepts that contain this attribute in their intents.

**Object hash table:** the key is the object identifier, and the entry for each object is the list of concepts that contain this object in their extents.

**Concept hash table:** the key is the concept number, and the entry for each concept contains its intent list, extent list, parent list and child list.

**OutPropList:** a list of attributes that do not satisfy the selection criteria.

**PropAncTable:** the key is the attribute identifier, and the entry for each attribute is a list of nearest ancestors that satisfy the criteria.

**newExtentTable:** the key is the attribute identifier. An entry exists for an attribute that is replacing one or more attributes that do not meet the selection criteria. Associated with the entry is a list of all objects which have been added as having this attribute. Also, associated with the entry is a list of all concepts which have had this attribute added to their intents.

At the very beginning, OutPropList is determined by running a for loop to check whether each of the attributes satisfy the selection criteria. Those not meeting the selection criteria are added to the OutPropList. This step would also be needed in the approach that simply recreates the formal context by removing those attributes from the original formal context and then rebuilding the new concept lattice from the new formal context. In the following performance
analysis of the algorithms to modify the concept lattice instead of rebuilding, \(|G| \) and \(|M| \) are the number of objects and attributes of the lattice, and \(|F_B| \) is the number of elements in the family (or number of concepts in current concept lattice), which is also the number of concepts in the lattice. The following process to modify the existing concept lattice using only those attributes satisfying the selection criteria has five main steps.

The first step creates the PropAncTable to store all attributes that do not satisfy criteria and their associated individual nearest ancestor attribute(s) that do satisfy the selection criteria. It examines every attribute in the OutPropList. The worst case scenario is that every attribute did not satisfy the selection criteria so that the outermost loop has the complexity of \(O(|M|) \). Then for every attribute, the algorithm needs to find its ancestor(s) that meets the selection criteria so that a fraction of all the attributes are involved, i.e., \(|M|/c \) attributes are checked. At this point, the complexity is \(O(|M|^2) \). Finally, in the innermost loop, the if block inserts the found ancestor into PropAncTable and newExtentTable. These operations have a constant complexity \(O(1) \). The else if block also has a complexity of \(O(1) \) because PropAncTable is implemented as a hashtable and its entry has all the closest ancestors that satisfy the selection criteria. Therefore, the complexity of the first step can be denoted by \(O(|M|^2) \).
The second step processes each attribute in the OutPropList. The task is to remove the attribute from all concepts' intents to which it belongs and then add into these intents its satisfying ancestor attribute(s). The newExtentTable entry for an ancestor attribute is updated by adding to the entry's extent the extents of the concepts whose intents had the ancestor attribute added, i.e., this ancestor attribute is now associated to the objects in the concept's extent. The conceptList for the newExtentTable entry of the ancestor attribute is also updated to include those concepts whose intents had the ancestor attribute added. The outermost loop has
complexity of $O(|M|)$ since in worst case, every attribute does not satisfy the selection criteria. Inside this loop, the algorithm needs to loop through the concept list to replace the attribute with its ancestors which matches the criteria. The worst case is that the attribute to be replaced is in the intent of every single concept. Assume the average number of ancestors found matching criteria for each attribute is $a$, then the complexity of the inner loop is $O(|F_B| \times a)$. Since $a$ is just a small constant, the complexity is just $O(|F_B|)$. Therefore, the complexity of the algorithm’s second step is $O(|M| \times |F_B|)$.

```plaintext
For each prop-i in OutPropList
{
    Find prop-i in PropAncTable
    For each c-j in conceptList for prop-i
    {
        Find c-j in conceptHashTable
        Remove prop-i from intent(c-j)
        Find prop-i in PropAncTable
        For each anc-k in Ancestors of prop-i in PropAncTable
            If anc-k not in intent(c-j)
            {
                Add anc-k to intent(c-j)
                Find anc-k in newExtentTable
                Add extent(c-j) to extent of anc-k in newExtentTable
                Add c-j to new concept-list of anc-k in newExtentTable
            }
    }
}
```

**Figure 5-2 Second Step of Algorithm to Create Granulated Concept Lattice**

The third step modifies the structure of the concept lattice starting from the bottom concept in it and going up one parent level at a time. It checks to see if the process of substitution of ancestor terms done in the second step causes the need for combining some of the modified concepts into one concept if they share the same intents or extents. First of all, the third step needs to loop through all the concepts by proceeding from the bottom upwards through the concept lattice. The complexity of the outermost loop is $O(|F_B|)$. The “while cur-p-list not empty”
block has complexity of $O(1)$, because the average number of parent concepts of each concept is a small constant. Then the following If, Else if, and Else blocks also have complexity of $O(1)$. If $P$ is not marked STOP, the algorithm first needs to cut off the edge between $P$ and $C$, and then loop through every parent of $P$ to add edges between each of them and $C$ if necessary. Again, since the average number of parent concepts for each concept is just a small constant, the complexity of this for loop is also $O(1)$. After that, the algorithm needs to update the parent of all children of $P$ to $C$ and also update the relationships between $C$ and $C_3$ as well if necessary. Similar to the previous loop, its complexity is also $O(1)$. Therefore, the overall complexity of the third step can be denoted by $O(|F_B|)$.

```
Add bottom-concept to c-list
While c-list not empty
{
    C = remove first element from c-list
    Find C in the ConceptHashTable
    cur-p-list = parent-list(C)
    While cur-p-list not empty
    {
        P = remove first element from cur-p-list
        Find P in ConceptHashTable
        If intent-list(C) == intent-list(P)
            Mark P for delete
        Else if extent-list(C) == extent-list(P)
            Mark P for delete
        Else if extent-list(C) subset-of extent-list(P)
            Mark STOP on P
        Else
        {   
            Add extent-list(C) to extent-list(P)
            if extent-list(P) == extent-list(C)
                Mark P for delet
        }
    }
}
```
If P NOT marked STOP on
{  
  If P marked delete
  {  
    Add extent-list(P) to extent-list(C)  
    updatePropertyObjectHashTable(P, C)  
    Remove P from parent-list(C)  
    For each P2 in parentList(P)
    {  
      Find P2 in conceptHashTable  
      Remove P from child-list(P2)  
      Add-edge = True  
      For each C2 in child-list(P2)
      {  
        Find C2 in conceptHashTable  
        If intent-list(C2) subset-of intent-list(C)  
        Add-edge = False  
      }  
      If add-edge and P2 NOT in parent-list(C)
      {  
        Add C to child-list(P2)  
        Add P2 to parent-list(C)  
        Add P2 to cur-p-list  
      }  
    }  
  }  
  For each C2 in childList(P)
  {  
    Find C2 in conceptHashTable  
    Add C to parent-list(C2)  
    Remove P from parentList(C2)  
    For each C3 in child-list(C)
    {  
      Find C3 in conceptHashTable  
      If intent-list(C2) ⊆ intent-list(C3)
      {  
        Remove C from parentList(C3)  
        Remove C3 from childList(C)  
      }  
    }  
  }  
}  
// If P marked delete  
Else add P to end of c-list  
}  
// If P NOT marked STOP on  
}  
// While cur-p-list not empty  
}  
// While c-list not empty

Figure 5-3 Third Step of Algorithm to Create Granulated Concept Lattice
The fourth step reconciles the information in the newExtentTable for an attribute with the information about that attribute in original PropertyHashTable. It checks every pair of concepts that contains a new attribute (ancestor attribute) to see whether the correct parent-relation or child-relation are taken care of. First of all, the complexity of looping through every attribute in newExtentTable is simply $O(|M|)$. Then the two loops of which each loops through the new concepts of prop-i has the complexity of $O(1)$ because the new concept brought in by each attribute is normally a very small constant. The following If block simply checks the extents of concept i and j, so its complexity is $O(1)$. Inside the else block, the double loop obviously has complexity of $O(|F_B|)$. So the overall complexity of this step is $O(|M| \times |F_B|)$.

```
For each prop-i in the newExtentTable
{
    For each c-j in newConceptList(prop-i) in newExtent Table
        For each c-k = c-j+1 to end newConceptList(prop-i) in newExtent
        checkRelationship(c-j, c-k)
    Find prop-i in PropertyHashTable
    If prop-i not in PropertyHashTable
    {
        If not extent of any existing concept j = extent of prop-i in newExtent Table
            Add prop-i to brandnewPropList
        }
    }
Else
{
    For each c-j in newConceptList(prop-i) in newExtent Table
        For each c-k in conceptList(prop-i) in PropertyHashTable and (c-k != c-j)
            checkRelationship(c-j, c-k)
}
```

Figure 5-4 Fourth Step of Algorithm to Create Granulated Concept Lattice

The last step determines if any new attributes were introduced in the modification process that did not already existing in the intent of a concept in the original concept lattice. If so, then a new
concept may need to be created such that it has as its intent just that new attribute and as its extent the extent associated with that attribute in the newExtentTable.

First of all, assume the average number of attributes brought into brandNewPropList by each existing attribute is a, then the worst case total number of attributes in brandNewPropList is \( a \times |M| \) so that the outermost loop has worst case complexity of \( O(|M|) \). Then the size of the newConceptList is based on the size of the concept lattice. The worst case is that the new attribute was added to the intent of every concept. The complexity of looping through the newConceptList is then \( O(|FB|) \). Inside this loop, the complexity of looping through the parentList of a particular concept is \( O(1) \), since the average number of parent concepts for each concept is a small constant. The overall complexity is \( O(|M| \times |FB|) \) at this point. Inside the following if block, besides the two for loops, all operations have complexity \( O(1) \). The first loop loops through all objects associated with of a particular concept. Its complexity is \( O(|G|) \) because on average a concept has \( |G|/2 \) objects. The second loop scans all new concepts of a particular attribute. We can also say the complexity of this block \( O(|FB|) \). Therefore, the overall complexity of this step is \( O(|M| \times |FB| + |M| \times \max(|G|, |FB|)) \). By simplifying it, the complexity can be denoted by \( O(|M| \times \max(|G|, |FB|)) \).
From these five steps, the complexity of the whole granulated concept lattice algorithm will be the maximum of $O(|M|^2)$, $O(|M| \times |F_B|)$, $O(|F_B|)$ and $O(|M| \times \max(|G|, |F_B|))$. Comparing to the result $O((|M|+|G|)\times |M|\times |F_B|)$ in (Qiao et al. 2003) which rebuilds the formal context table, the above
granulated concept lattice algorithm works more efficiently if the lattice contains large number of concepts (meaning that $|F_B|$ is much larger than $|G|$ and $|M|$)—which is a very possible case in Gene Ontology. But when the new formal context or the concept number is quite small, it can be simpler to just rebuild and redo the concept lattice using Qiao’s algorithm.

5.3 Concept Approximation Querying

The objective of concept approximation is to determine the concept that best agrees with typically a set of objects, a set of properties or a pair of inputs: a set of objects and a set of attributes. The standard approaches to concept approximation do not use similarity assessment to measure agreement but instead use approximation operators to find the formal concept(s) that represent the best solution to the inputs. The algorithms for concept approximation are all developed based on the presentation in section 4.5. All three approaches to concept approximation have been implemented: Saquer’s (discussed in section 4.5.1), Hu’s (in section 4.5.2) and Yao’s (in section 4.5.3). Concept approximation has been implemented so that it can be performed for a set of objects or a set of attribute.

This functionality has been implemented as a query capability that permits the user to provide the desired input and then produces the lower and upper approximation for the entered set of objects and the associated concepts for those approximations. Similarly, it produces the lower and upper approximation for the entered set of properties and the associated concepts for those approximations.

5.4 Concept Similarity Assessment
Tversky’s set-theoretic similarity approach described in section 4.6.1 serves as the basis for assessing concept similarity. One additional consideration, however, is that for annotated objects, the properties in the intent of a formal concept may exist in an ontology such as the Gene Ontology. The method, therefore, used in determining intent component similarity of the formal concepts incorporates the ontological or semantic similarity between the properties in the two intents. The following equation defines the measure of similarity between two formal concepts X and Y that is used in the software developed for this thesis:

\[
S_{c-SS-Tversky}(X,Y) = \omega \frac{|E(X) \cap E(Y)|}{r} + (1 - \omega) SS_{PF-max-weighted}(I(X),I(Y))
\]

The formal concept similarity measure used in this thesis software generalizes the measurement of concept similarity back to the Tversky model by incorporating the weighted average of the maximum pairwise similarities between two sets of properties \(X_p\) and \(Y_p\) expressed as

\[
S_{PF-max-weighted}(X_p, Y_p) = \frac{\alpha \sum_{i=1}^{nX} \max_{j=1}^{nY} SS(x_i, y_j) + \beta \sum_{j=1}^{nY} \max_{i=1}^{nX} SS(x_i, y_j)}{\alpha nX + \beta nY}
\]

For simple averaging of the two separate maximum totals, \(\alpha=1\). In the above equation \(SS\) represents any semantic similarity measure between terms within the ontological terminology. The \(x_i\) and \(y_j\) represent the terms in the two property sets, i.e., the intents of \(X\) and \(Y\), respectively, and \(nX = |X_p|\) and \(nY = |Y_p|\). This approach allows the user to select between an inclusion measure or a similarity measure for measuring extent similarity and intent similarity. In addition, for the inclusion measure, the user can explicitly use a reference concept or allow reference to be determined based on the maximum intent size and maximum extent size. To summarize, there are three options: the max option, where \(r\) becomes \(\max(|E(X)|, |E(Y)|)\); the union option where \(r\) becomes \(|E(X) \cup E(Y)|\); and the reference option, where the second concept, the concept \(Y\), is the reference concept, so that \(r\) becomes \(|E(Y)|\). For the reference option, determining the intent similarity uses the \(S_{PF-max-ave}\) equation with \(\alpha=1\) and \(\beta=0\) since \(Y\) is serving at the reference concept and for each property \(X_p\), the maximum similarity is found across all the attributes in \(Y_p\). For the max option, \(\alpha=1\) and \(\beta=0\) if \(|Y_p| > |X_p|\), otherwise \(\alpha=0\) and \(\beta=1\) since the concept with the maximum intent size serves as the reference concept. For the union option, \(\alpha=\beta=1\) since no
concept serves as a reference so that the simple average of the two maximums is used. Another separate option allows the user to select from several different ontological or semantic similarity measures to be used for the SS function in the assessment of concept similarity between formal concepts whose intents consist of terms from an ontological terminology.

5.5 Concept Lattice Similarity Assessment

There are two methods in concept lattice similarity: assessing the concept similarity within each concept lattice, and the ordering of concepts within each concept lattice. In order to do both of these operations, a mapping between the concepts in the first concept lattice to the concepts in the second concept lattice needs to be established: every concept Ci in the first concept lattice should be mapped to a concept Cj whose concept similarity with Ci is the greatest one in the second concept lattice. There is also a threshold value which can be set between 0 and 1. Thus, only when the similarity value is greater than the threshold value, it is considered that there is a mapping between the two concepts. But if two or more concepts in the second concept lattice map to a concept in the first concept lattice with the same similarity values, randomly keep one mapping from them.

Assume the concept lattice similarity between concept lattice CL1 and concept lattice CL2 is to be determined. First concept similarity can be used between the formal concepts in CL1 and CL2 to establish mappings from all concepts in concept lattice CL1 to a subset of concepts in concept lattice CL2 because two or more concepts in CL1 can be mapped to one same concept in CL2. To determine the concept similarity, just use the concept similarity assessment described in section 5.4. When the concept lattices are from different domain having no overlap on the extents, the concept similarity will be based on only the intent similarity. Now assume that based on concept similarity measurement, CL1 contains concepts B, E, and F and CL2 contains concepts J, I, K where concept B in CL 1 mapped to concept J in CL 2, E in CL 1
to I in CL 2, and F in CL 1 to K in CL 2. Then concept lattice similarity can be calculated based on how distance preserving this mapping is. Then for every pair of concepts in CL1: (B, E), (E, F), and (B, F), find out the concept similarity value. Also find out the concept similarity for every corresponding pair of concepts in CL2: (J, I), (I, K), and (J, K). Next, calculate the difference between the concept similarity of each pair in CL1 and the concept similarity of its corresponding pair in CL2. Finally, the total of the differences between the similarity pairs is averaged and subtracted from 1. Using this simple example the similarity mapping will be:

\[ \text{simMap}(\text{CL1}, \text{CL2}) = 1 - \frac{\text{abs}(\text{SS}(B, E) - \text{SS}(J, I)) + \text{abs}(\text{SS}(E, F) - \text{SS}(I, K)) + \text{abs}(\text{SS}(B, F) - \text{SS}(J, K))}{3} \]

The other method is assessing the difference in ordering of concepts within each concept lattice. An order function is defined as follows:

\[ \text{order}(X, Y) = \begin{cases} 
1 & \text{if } X \text{ is the ancestor of } Y \\
-1 & \text{if } Y \text{ is the ancestor of } X \\
0 & \text{if no ancestor relationship exists between } X \text{ and } Y 
\end{cases} \]

Then the ordering similarity difference is defined as

\[ \text{ordMAP}(\text{CL1}, \text{CL2}) = 1 - \frac{\text{abs}(\text{ord}(B, E) - \text{ord}(J, I)) + \text{abs}(\text{ord}(E, F) - \text{ord}(I, K)) + \text{abs}(\text{ord}(B, F) - \text{ord}(J, K))}{3} \]

These two methods of concept lattice similarity provide the user with an assessment of how well the mapping between the concepts from one concept lattice to another concept lattice preserve either the concept similarity or the ordering of the concepts within the concept lattice hierarchy.

6 User Interface Design

In order to provide these concept lattice analysis capabilities a user–friendly interface is provided and allows numerous options for creating concept lattices based on a specified ontology and an associated annotation file. The user interface includes the following tabs: “Concept Lattice”,

50
“Granulation & Filtering”, “CL (concept lattice) similarity” and “Approximation”. It basically uses the same style of other tabs of the QUOTA system since it is integrated into this existing system. This chapter presents the user interface that extends and expands the QUOTA system. It describes each part of the user interface and explains features and capabilities of the system. The overall user interface with the tabs is shown in Figure 6-1. The first five panels are from original QUOTA, and the last four new panels will be explained in the following sections.

![Overall User Interface of QUOTA](image)

Figure 6-1 Overall User Interface of QUOTA

6.1 Concept Lattice Panel

To meet the needs of capturing different views of the data, several different kinds of concept lattices can be produced. The following basic concept lattices are under “Selected Concept Lattice” option of the “Concept Lattice” tab:

1) A general concept lattice,
2) Object oriented concept lattice (OOCL),
3) Property oriented concept lattice (POCL),
4) The complement of the general concept lattice.

For each kind of concept lattice, the user can choose the following options for viewing the content of the nodes:
   a. intent only for each formal concept,
   b. extent only for each formal concept,
   c. both the intent and the extent for each formal concept.

![Figure 6-2 User Interface of Concept Lattice Panel](image)

The system provides two ways to input the selected annotated objects. The user may choose desired annotated objects from the list on the left hand side of the UI shown in figure 6-2 and click the “Add” button below. This method is how annotated objects were selected in the original QUOTA system.

With this input method, the objects that the user adds are displayed in “Selected Annotated Objects” text box. The problem, however, with this method is that too many annotated objects may exist for the user to choose from and add manually. To address this
problem, another input method allows the user to click the browse button to select a file on the host computer as input.

This method is a much more practical way to use the software. One format of the selected file is listing attribute names in the first line with each one separated by space, and listing one object name and its relation with all above attributes using 1 or 0 in each following line, which is just like a formal context tale. The other format of the selected file is only one annotated object name (gene name) in each line, and the benefit is that the user does not need to list gene IDs since the software can automatically transfer gene names to their IDs based on a file containing all gene names with their IDs which can be downloaded from ftp://ftp.geneontology.org.

After the user selects the input option and the desired kind of concept lattice, and what information is to be displayed, for example, intents only, the user then clicks the “Create Concept Lattice” button. The software then produces a report of the concepts with their associated information as determined by the user (all intents or/and extents of the concepts) and the edges that exist between the concepts in the concept lattice. This report is displayed in the “Result” text box. The results may be saved to a file in the QUOTA workspace folder by clicking “Save Results to File” button. Figure 6-3 shows the result of the creation of a concept lattice.
Clicking the “Get Visualization of Concept Lattice” button, the software produces a GML file and automatically calls VTK software to produce a 3D visualization of the concept lattice. This visualization is provided so that the user can understand the concept lattice. To finish and clear all results, the user simply clicks the “Clear” button. The use of get result button and clear button are the same for the later two tabs.

6.2 Granulation and Filtering Panel

The “Concept Lattice” panel provides the user with various kinds of concept lattices that are built using only the terms (attributes) that directly annotate the objects. No additional information is used about the structure of the ontological terminology from which the terms come. Granulated concept lattices, however, can be created if the structure of the ontological
terminology is used since the terms (attributes) exist in a hierarchical structure from more
general terms to more specific terms. Granulation occurs when a concept lattice is built using
terms that are more general than but hierarchically related to the terms actually used to annotate
the objects but that do not meet the user-specified selection criteria.

This feature assumes the existence of an already created concept lattice file which is
referred to here as the original concept lattice. Figure 6-4 shows the “Granulation & Filtering”
panel. The user enters an existing concept lattice file and then completes the Selection field by
choosing to either “filter” or “granulate” the attributes based on certain characteristics of the
attributes as determined by the structure of the ontological terminology. The Selection Criteria
field then allows the user to specify what characteristics of the term are to be used. The currently
available characteristics or metrics for a term are the term’s information content (IC) value,
interval rank (IR) value or its level based on the hierarchical structure of the ontology.

As previously explained the difference between “filtering” and “granulation” is that in
filtering, only direct annotations not meeting the selection criteria are removed and the original
concept lattice is modified without rebuilding a new concept lattice and in granulating the direct
annotations not meeting the selection criteria are not simply removed but are instead replaced by
their most immediate ancestor term meeting the selection criteria. The original concept lattice is
modified based on these replacements.
The user may create several filtered or granulated concept lattices from the same original concept lattice using different selection criteria, and then the concept lattice similarity assessment capability can be used to provide information on how the concept lattice has changed based on modifying the filtering parameters. This capability helps the users identify certain relationships between the annotated objects more readily. Finally the user can save results of granulated or filtered concept lattices as well.

6.3 Concept Lattice Similarity Panel

The “CL Similarity” panel can be used to determine the similarity between two sets of annotated objects. The user can first create a separate concept lattice for each set of annotated objects and then use the CL Similarity panel to determine the similarity between these two concept lattices. Figure 6-5 shows the panel for assessing similarity between two concepts. The user first enters the two concept lattice files previously created by either the “Concept Lattice” or “Granulation & Filtering” panels.
To assess the similarity between two concept lattices, both the similarity between individual concepts from the two difference concept lattices must be determined as well as the similarity in the structure of the two concept lattices. Similarity between individual concepts is determined as

\[ w \times \text{extent similarity} + (1 - w) \times \text{intent similarity}, \]

The \( w \) is in \([0, 1]\) and serves to aggregate the extent similarity and the intent similarity between two individual concepts. This weight is set by the user to enrich the flexibility of the method by allowing concept similarity to be based entirely on extent, entirely on intent or as a weighted average of the two. Also if simMAP is selected for the similarity calculation method, the weight \( w \) used in simMAP will be the same as the \( w \) set up here.

The extent and intent similarity are determined as explained previously in section 5.5. An important aspect in determining these similarity values is the normalization factor for the
amount of overlap or intersection between the two extents or the two intents. The user selects
the normalization factor as max, union and reference.

The Threshold value allows the user to specify a minimum value for concept similarity so
that only concept similarities greater than or equal to this threshold value are used to calculate
similarity between the concept lattices. If all concept similarity values are to be included, the
threshold value should be set to 0.

The similarity calculation method allows the user to select between simMAP and
ordMAP concept lattice similarity assessments when calculating concept lattice similarity. These
two ways are discussed in section 5.5.

6.4 Query Panel

The “Query” panel provides a series of useful queries that may be done after reading in concept
lattice file created in the “Concept Lattice” or “Granulation & Filtering” panel. Five kinds of
queries are available. First, by entering its concept ID the user can retrieve the extent, intent,
parent concepts, children concepts and sibling concepts of any concept in the concept lattice.
Second, the user, by providing two or more concept IDs, can retrieve their join concept or meet
concept in the concept lattice. Third, the join-irreducible concepts and the meet-irreducible
concepts are retrieved into the text box by the user simply clicking the buttons in lower right
corner. Fourth, after the concept lattice is loaded, the equivalence classes of all objects/attributes
will be provided automatically along with the associated attributes/objects with each equivalence
class in the lower right corner of the UI. Fifth, the user may provide a set of objects or a set
attributes, and retrieve for the concept lattice the upper and lower concept approximation
concepts. These can be calculated using the three different methods: Saquer’s, Yao’s and Hu’s
method which are discussed in section 4.5. The approximation result lists both the concept ID
and the intent and extent of the approximating concept.
Figure 6-6 User Interface of Query Panel

This query panel offers the user a convenient and easy method of finding information about the concept lattice. The user does not need to examine the concept lattice file or look at a complicated lattice visualization image to answer specific questions about the concept lattice. These queries can provide results much easier and faster.

7 Evaluation

This chapter presents the validation and evaluation of the new functionality added to the QUOTA system as part of this thesis. This evaluation is based on several illustrative data sets used to create a variety of concept lattices and also on exercising various user interface options to demonstrate the added functionality. These examples include small sample data taken from various research papers as well as real world data which includes a breast cancer gene lists from the University of Cincinnati’s Children’ Hospital and Medical Center (UCCHMC), a well-known
set of gene products GPD194 (Popescu et al. 2006), and a set of proteins that have been used to validate semantic similarity measures (Pesquita et al. 2008). These data are discussed in each section where they are introduced. This evaluation demonstrates that the software is useful for gaining insights into the data sets and "discovering knowledge" about the relationships and interactions between the data.

7.1 Variety of Concept Lattice

The following concept lattices are constructed using the formal context example given in (Yao 2004). The following figures illustrate the testing of the implemented algorithms to produce the four different kinds of concept lattices. These results agree with those in (Yao 2004) and are shown in Figure 3-1. In the figure, the graph of the concept lattice is produced by VTK taking as input a gml file produced by the concept lattice creation software. The listings of the formal concepts shown to the right of the following figures are produced by the lattice creation software and are displayed to the right of the visualization. These listings are part of the user interface displayed in figure 6.3 and the text output file created by the concept lattice creation software. Each of the four kinds of concept lattice is presented below:

2) The **general concept lattice** is shown in figure 7-1 with both extents and intents of the formal concepts listed to right.
In the right part of Figure 7-1, the first 9 lines (0-8) give a numeric identification to the formal concept and identify the extents and intents for each formal concept. Each concept is represented by a number from 0 to 8, and their extents and intents respectively are divided by a semicolon. For example, concept 8 has two objects 1 and 6 in its extent and two attributes a and e in its intent. After the 9 concepts, the structure of the concept lattice is represented by a set of edges between these concepts. For example, the bottom line “8: 3; 6;” means that concept 8 is the parent concept and is connected with its children concepts 3 and 6.

2) The object oriented concept lattice (OOCL): is shown in Figure 7-2.
3) The **property oriented concept lattice (POCL)**: is shown in Figure 7-3.

---

**Figure 7-2** Result of OOCL Given by the Software

**Figure 7-3** Result of Concept POCL by the Software
4) The **complement of the general concept lattice** is shown in Figure 7-4.

![Figure 7-4 Result of Complement Concept Lattice Given by the Software](image)

This one example taken from *(Yao 2004)* is presented to illustrate the functionality of creating the four kinds of concept lattices. Numerous other examples of formal contexts have been used to verify the correctness of the concept lattice creation software.

### 7.2 Granulation and Filtering of Concept Lattice

This section shows how the granulation and filtering functionality of the software works using example data for a set of genes that are annotated by terms from the Gene Ontology. This example is taken from *(The Gene Ontology Consortium 2000)*. Figure 2-2 shows the relevant portion of the BP branch of the GO. The genes being annotated by GO terms in the hierarchy are taken from three species. For example, in this sample data, the term “DNA litigation” annotates one gene from the yeast species (blue), two genes from the fly species (purple) and two genes from the rat species (red). The links between nodes indicate “is-a” relations where as one goes
lower in the hierarchy, the associated terms with the nodes becomes more specific. Therefore, the genes annotated by (attached to) a lower node in the GO hierarchy can be considered by default as annotated by the node’s ancestors and, therefore, indirectly annotated by those terms. One, however, can also just consider the terms that are directly annotating a gene and not the ancestor terms. That is, one creates a formal context without considering the hierarchy and then forms its concept lattice using only the direct annotations as the attributes for the gene objects. Figure 7-5 shows this result.

![Figure 7-5 Concept Lattice of Figure 2-2](image)

For this example data, there are 43 unique genes and 10 direct annotations, i.e. a 43 by 10 formal context. One can see that this formal context produces 20 concepts in the lattice using only the direct annotations, and figure 7-6 shows the list of concepts and their intents and extents.
Figure 7-6 List of Concepts of Figure 7-5

Now this example is based on a small formal context, but imagine a much more complicated case containing more genes and annotations. The resulting concept lattice can be extremely large and difficult to see clearly. The user has the ability to select the degree of granulation for which direct annotations and indirect annotations are to be included in the concept lattice. The degree of granulation as explained in section 5.3.1 can be specified by the level, the interval rank or the information content of the term within the ontology. This example uses only a small portion of the GO hierarchy and assumes that 'DNA metabolism' is at level 0 to illustrate granulation based on using the level option. To granulate the concept lattice shown in Figures 7-5 and 7-6, level 2 has been selected. This granulation results in the inclusion of all direct annotations from the top level to level 2. All direct annotations below level 2 are moved up to all of their ancestors at level 2. These indirect annotations at level 2 are also included with the direct annotations to form the set of attributes used in creating the concept lattice. With this granulation specified, the formal context results in the 43 genes and the number of annotations is 5, a 50% reduction in the number of attributes. Typically the number of annotations is reduced, but the most important reduction is in the number of formal concepts produced from the formal context because more
genes “share” the same annotation when the granulation of the annotations is coarser, i.e., the annotations at levels below the specified granulation level are replaced with ancestors that meet the granulation level. These ancestors are more general terms. The resulting granulated concept lattice at level 0 through level 2 in the GO is shown in figure 7-7. Figure 7-8 shows its list of concepts.
Compared to figure 7-5, this new concept lattice contains only 12 or 60% of the number of concepts in figure 7-5. Some concepts remain exactly the same when granulation is performed on the original concept lattice. For example, concepts 0, 4, 6, 8 and 9 in figure 7-6 are exactly the same as concepts 0, 3, 5, 8, and 9 in figure 7-8 respectively. That is, approximately 42% remain the same. These concepts remain the same because all the attributes in the intents for these concepts are terms that were directly annotated in Figure 2-2 and are at level 2 or higher in the hierarchy and also no lower direct annotations for genes needed to be replaced by indirect annotations using these terms.

The reduction in the number of concepts in the granulated concept lattice can easily be seen to occur because the one term DNA_Dep_DNA_replication at level 2 is substituted for all the level 3 or greater terms annotating genes for which it is an ancestor term. Because of this substitution, concept 1 in Figure 7-8 has an extent consisting of those 34 genes and replaces 8 concepts in Figure 7-6: 1, 10, 12, 13, 15, 17, 18, and 19. Some concepts in Figure 7-8 have exactly the same extents as those in Figure 7-5, but their intents differ, for example, concepts 2, 11, 4, 7, and 10 have the same extents as concepts 2, 3, 5, 7 and 11 respectively in Figure 7-6. That is, approximately 42% of the granulated concepts have the exact same extents but different
intents. The difference in their intents is due to the substitution of ancestor terms that meet the level 2 specifications for those direct annotations that occur at level 2 or higher. Thus only two concepts or approximately 16% differ in both intent and extent. Concept 1 in the granulated concept lattice is the one that varies the most since its replaces 8 different concepts in the original concept lattice. Concept 6 also differs in both intent and extent from any of the original concepts.

The previous example using a sample portion of the GO annotating a small set of genes is meant to illustrate how concept granulation works. Another real-world data set: the breast cancer data from UCCHMC contains two sets of genes: the training set containing a set of genes already known as cancer-causing and a test set containing another set of genes to be compared to the training set to see whether any of the genes in the test set could be strongly related to the breast cancer causing genes. These two sets only contain lists of gene names. Because of the data file format, it was necessary to write additional program code to compare each gene name with the contents of GeneName2ID.txt (www.geneontology.com) file in order to determine each gene’s numeric unique gene identification. The annotations used to create the concept lattice for these sets of gene products are found in the “gene2go” file (www.geneontology.com). For the training set, there are 15 genes and 167 associated annotations can be found in “gene2go” file. Figure 7-9 shows the concept lattices of the training set with 59 concepts. The formal concepts which are associated with the visualization are listed in the appendix at the end of the thesis.
This real-world training set annotation file demonstrates that although the objective of granulation which uses more general ontology terms is to reduce the complexity of the concept lattice, complexity reduction does not always occur. The reason in this case is that the whole GO term hierarchy contains over 20 levels. But most of the GO terms are located at certain levels like level 6 or 7. Only a few of them are at higher or lower levels. The higher (or lower) levels where granulation may be chosen level compared to level 6 where most of the annotations are taken from, the fewer the number of GO annotations existing at these levels. Mapping lower level GO terms to higher levels can in some instances reduce the number of attributes in the concept lattice. This occurs when multiple annotating terms all replaced by the same ancestor term. In other instances the number of attributes can greatly increase when lower level annotating terms instead of many of them mapping to the same ancestor term, they all map to their own ancestor and many also map to more than one satisfying ancestor since the GO hierarchy has multi-inheritance. For example, the original concept lattice of the training set contains GO terms that are as low as level 12. Thus for experimentation purposes, the granulation levels are set to about 1/2 and 1/4 of this deepest level which produce level 6 and
level 3, respectively. Figure 7-10 and figure 7-11 show the concept lattice of the training set granulated to level 6 and level 3. The formal concepts which are associated with each visualization are given in the appendix.

Figure 7-10  Concept Lattice of Training Set Granulated to Level 6
After granulation to level 6 and level 3, the numbers of concepts increase from the original 59 to 111 and 69, respectively. However, granulation is still useful in this example as some genes might become much more similar at a higher level of granulation and groups more genes together with the granulated concept lattice. The concept similarity procedure which is described below in section 7.3 is used on all the pairs of genes in the training set for the original and the granulated concept lattices. The following table shows the similarity results at 0.1 increments of similarity. For example, only 8.6% of the gene pairs have a similarity value of 0.5 or higher.

<table>
<thead>
<tr>
<th>similarity value range</th>
<th>original concept lattice</th>
<th>granulated concept lattice to level 6</th>
<th>granulated concept lattice to level 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 - 0.1</td>
<td>4.8%</td>
<td>5.7%</td>
<td>0.0%</td>
</tr>
<tr>
<td>0.1 - 0.2</td>
<td>21.0%</td>
<td>9.5%</td>
<td>3.8%</td>
</tr>
<tr>
<td>0.2 - 0.3</td>
<td>17.1%</td>
<td>14.3%</td>
<td>1.9%</td>
</tr>
<tr>
<td>0.3 - 0.4</td>
<td>19.0%</td>
<td>15.2%</td>
<td>16.2%</td>
</tr>
</tbody>
</table>
Table 7-1  Similarity Results for Original CL and Granulated CLs of the Training Set

<table>
<thead>
<tr>
<th>Interval</th>
<th>0.4 - 0.5</th>
<th>0.5 - 0.6</th>
<th>0.6 - 0.7</th>
<th>0.7 - 0.8</th>
<th>0.8 - 0.9</th>
<th>0.9 - 1.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Value</td>
<td>29.5%</td>
<td>24.8%</td>
<td>1.9%</td>
<td>6.7%</td>
<td>24.8%</td>
<td>12.4%</td>
</tr>
<tr>
<td></td>
<td>0.0%</td>
<td>5.7%</td>
<td>23.8%</td>
<td>0.0%</td>
<td>0.0%</td>
<td>33.3%</td>
</tr>
<tr>
<td></td>
<td>0.0%</td>
<td>0.0%</td>
<td>6.7%</td>
<td>0.0%</td>
<td>0.0%</td>
<td>0.0%</td>
</tr>
</tbody>
</table>

Table 7-1 shows that in the original concept lattice of the training set, most pair of genes having a similarity value under 0.5. After granulation to level 6, the similarity results that are between 0.5 and 0.6 largely increase from 6.7% to 24.8%, also 5.7% of pairs of genes having a similarity between 0.6 and 0.7 compared to 1.9% in the original concept lattice indicates that some genes become more similar after granulation. The similarity value of each pair of genes gets even greater after granulate the concept lattice to level 3 as totally 40% of pairs have similarity results between 0.7 and 0.9. As the annotating terms are generalized the genes have more similar attributes between each other after granulation. Though granulation cannot always simplify the concept lattice, it can be used to generalize the description of the genes to see at what GO term levels they become more similar. Experimenting with granulation of the annotation file might also help to identify where the descriptions of the set of genes begins to diverge.

Using this same training set an example of IC granulation is presented. The information content of a node in a hierarchy is given as:

\[ IC(n) = \log \left[ \frac{(hypo(n) + 1)/max}{\log (1/ max)} \right] = 1 - \log (hypo(n) +1)/\log (max) \]

where hypo(n) is the number of descendents of node n, and max is the maximum number of nodes in the hierarchy. The IC value of a leaf node is 1 because it has no descendants. As one proceeds up the hierarchy to the root node, the IC value decreases gradually to 0. Figure 7-12 shows the granulated concept lattice when IC criteria set to 0.25 and for each attribute that has IC
value in GO hierarchy greater than 0.25 will be granulated to its nearest satisfying ancestors. Figure 7-13 shows the extents and intents of each concept in figure 7-12.

Figure 7-12 Granulated Concept Lattice on IC/IR Values of Figure 2-2

Figure 7-13 List of Concepts of Figure 7-12

With this granulation specified, the formal context results in the 15 genes and the number of annotations is 16, more than 90% reduction in the number of attributes. Number of concepts reduces from 59 to 15, a 75% reduction. Figure 7-14 is a screenshot of using the equivalence class query in Query panel in QUOTA software to calculate the equivalence classes and their...
associated GO terms of the above granulated concept lattice. The result shows that there are only 10 equivalence classes compared to 15 in original concept lattice which also indicates that the annotating terms are better generalized and the genes have more similar attributes between each other after granulation.

7.3 Concept Lattice Similarity

The GPD 194 data set contains 194 human gene products (proteins) that are classified into three protein families: myotubularin, receptor precursor and collagen alpha chain. These three protein families map to 27 different genes. The characteristics of each family are shown in Table 7-2. This data set was selected for experimenting with the concept lattice similarity functionality based on the rationale provided for its use in a previous experiment (Popescu et al. 2006). Multiple genes that are well-characterized by GO annotations are associated with each family. Although these families are considered distinct from one another, they may be considered similar.
based on sharing similar functions as described by higher-level GO terms. An important consideration for using this data set is that these families had already been determined through the Markov clustering (MCL) capability of ENSEMBL (http://www.ensembl.org) and, therefore, provide a benchmark data set to use for experimentation.

Table 7-2 Characteristics of the GPD194 Data Set [Popescu et al. 2006]

<table>
<thead>
<tr>
<th>ENSEMBL Family (ENSF)</th>
<th>$F_i =$ Protein Family</th>
<th>No. of Genes</th>
<th>Gene Symbols (in order)</th>
<th>$N_i =$ No. of Sequences</th>
</tr>
</thead>
<tbody>
<tr>
<td>335</td>
<td>myotubularin</td>
<td>7</td>
<td>MTMR1=4, MTMR6=8</td>
<td>21</td>
</tr>
<tr>
<td>73</td>
<td>receptor</td>
<td>7</td>
<td>FGFR1=4, RET, TEK, TIE1</td>
<td>87</td>
</tr>
<tr>
<td>42</td>
<td>precursor collagen alpha chain</td>
<td>13</td>
<td>COL1A2, COL2A1, COL24A1, COL27A1, COL2A1, COL3A1, COL4A1, COL4A2, COL4A3, COL4A6, COL5A3, COL9A1, COL9A2</td>
<td>86</td>
</tr>
</tbody>
</table>

In the first experiment, the annotation file for each family is used to build a separate concept lattice for each family and then the combined annotation file is used to build one concept lattice for all three families. The purpose of this experiment is to determine a degree of overall similarity between each pair of families and between each family concept lattice and the one combined concept lattice. Determining concept lattice similarity consists of calculating the concept similarity between individual concepts and then the structure similarity between the two concept lattices. While calculating the concept similarity, several parameters need to be given by users as discussed in the user interface section 6.3, like the threshold value, the similarity calculation method and the $w$ parameter which adjusts the weight of extent similarity and intent similarity in determining overall concept similarity.

Table 7-3 shows the result using genes as object in concept lattice when the weight parameter $w$ set to 0.5, and the threshold value set to 0. The three options are for “Normalization Factor” when calculating similarities. Also the simMAP method is selected for the concept lattice similarity assessment.
From above we can see that for this particular example, the highest similarity values are produce under option 2. From the table, one can also see that concept lattice similarity is not symmetric because the concept lattices in columns are always used to provide the best mapping for the concept lattices in rows. Therefore, if a concept $c_i$ in Myotubularin row maps to the concept $c_j$ in Receptor column, $c_j$ may not also map to $c_i$ when we are doing Receptor in rows and Myotubularin in columns instead because there can be another concept in Myotubularin concept lattice having a greater similarity than $c_i$ with $c_j$. One can see from the table that for all options the similarity values in row “GPD194” are always smaller than all the others. This result occurs because the overall concept lattice is being compared to each individual family concept lattice which provides best mappings. For all three options, the individual collagen family concept lattice has the highest concept lattice similarity to the GPD194 concept lattice. This result fits since this family has the biggest concept lattice and therefore covers more of the GPD194 concept lattice. Similarly, one can see from the table that for all options the similarity values in column “GPD194” are always greater than others. This result occurs because concepts in each
concept lattice for an individual family can find best matching concepts in GPD194 concept lattice with higher similarities, or even similar concept lattice structure.

The semantic similarities between GO terms used in all examples in this thesis were not actually calculated by QUOTA system, but were received from Sriram Ramakrishnan, a graduate student in Miami University 09’, and were then put into a database so that they can be directly read in when calculating concept similarities.

From the individual concept lattices of three protein families shown in figures 7-15-a, 7-15-b, and 7-15-c and their combined one (GPD194) shown in figure 7-16, one can see that a set of concepts (sub-concept lattice) in the GPD194 concept lattice corresponds to each individual one. This visualization further explains why concepts in each individual concept lattice can always find a highly similar concept in the GPD194 one. The extent and intent of each concept in the following four concept lattices are in the appendix.

![Figure 7-15-a Concept Lattice for Myotubularin Family using Gene Objects](image)

**Figure 7-15-a Concept Lattice for Myotubularin Family using Gene Objects**
Figure 7-15-b  Concept Lattice for Receptor Family using Gene Objects

Figure 7-15-c  Concept lattice for collagen family using gene objects
Figure 7-16 Concept Lattice of GPD194 Data using Gene Objects

Figure 7-16 uses red, yellow and blue colors to represent the myotubularin, receptor, and collagen respectively concepts, but also uses orange, green and purple to represent the concepts within the GPD194 concept lattice which include genes from two different families. The white node is the top concept which includes all the genes from the three families and the black node is the bottom concept which includes an empty set of genes. To visually illustrate that the GPD194 combined concept lattice reveals the three families, a coloring scheme with the high-contrast colors, red, yellow and blue was used to easily distinguish the three families. Concepts for which the extent consists of only genes in the myotubularin family are colored red. Concepts for which the extent consists of only genes in the receptor family are colored yellow. Concepts for which the extent consists of only genes in the collagen family are colored blue. For a concept whose extent includes genes from two or three families, we use the mixed color of those
involved families. The orange color is a mixed color of red (myotubularin) and yellow (receptor), so an orange concept, like node 38, has an extent with genes from the myotubularin and receptor families. Likewise, a green concept, like node 6, has an extent with genes from the receptor (yellow) and collagen (blue) families. A purple concept, like node 26 has an extent with genes from the myotubularin (red) and collagen (blue) families. The only concept that consists of genes from all three families is the top node.

In figure 7-16 the 9 concepts colored orange, green and purple identify the connecting concepts between the three families and indicate an overlap in annotations that are relevant to both families. For instance, the concept with ID 8 in GPD194 concept lattice and the concept with ID 7 in collagen concept lattice have the following genes and GO terms:

Concept #8 (GPD194):
- GO terms: GO_0007155;
- Genes: COL21A1, COL24A1, COL4A3, COL4A6, COL5A3, COL9A1, RET

Concept #7 (Collagen):
- GO terms: GO_0007155;
- Genes: COL21A1, COL24A1, COL4A3, COL4A6, COL5A3, COL9A1

The only difference between those two concepts is the additional RET gene in GPD194. These connecting concepts can be used to identify the commonality between the pairs of protein families.

FCA analysis on the GPD194 produces nine top level formal concepts. These top level concepts can be further analyzed to determine the clustering of genes or the families existing in this annotation file. Each of the top level concepts represents an equivalence class over the attributes, i.e., the annotations. These equivalence classes identify all attributes that annotate the exact same set of objects. Ordering these concepts from largest to smallest based on the size of the extents produces concepts 1 with 12 objects, 8 with 7 objects, 37 with 7 objects, 49 with 7 objects, 4 with 5 objects, 38 with 4 objects, 6 with 3 objects. The following describes a process that was used to easily identify the three families in the combined concept lattice. This example
and process has not been implemented as an algorithm but is considered for future research work.

To begin the analysis process, one creates a potential family group for each of these nine formal concepts. Further analysis shows that concept 1 has 5 overlapping extent objects with concept 8, 3 overlapping extent objects with concept 4, and 1 overlapping extent object with concept 26. It appears that some of the objects in these concepts might be more strongly associated with the objects in family concept 1. The annotation similarity of the overlapping objects in each of those concepts can be compared to those objects in the extent of concept 1 and to the other objects in each of their respective extents. In each overlapping set, the objects in the set are more similar to those in family concept 1, so they are considered to be part of family 1 concept and removed from their respective family 8 concept, family 4 concept, and family 6 concept. The process is then repeated with family 37 concept which is the next largest family since family 8 concepts had 5 objects removed.

Concept 37 has 3 overlapping extent objects with concept 38 and 1 overlapping extent object with concept 42. In each overlapping set, the objects in the set are more similar to those in concept 37, so they are considered to be part of family 37 concept and removed from their respective family 38 concept, and family 42 concept. Now concept 49 has 2 overlapping extent objects with concept 4, 1 overlapping extent object with concept 38, 2 overlapping extent objects with concept 6, 1 overlapping extent object with concept 8, and 1 overlapping extent object with concept 42. Again these overlapping objects are more similar to those in concept 49 and are considered part of family 49 concept and removed from their respective family 4 concept, family 38 concept, family 6 concept, family 8 concept, and family 42 concept. At this point, only family concepts 1, 37, 49, 6 and 8 have objects in them. Family concepts 6 and 8 have exactly 1 object COL21A1 and could be considered a separate family but assessing the similarity of this object to the objects in the other family concepts 1, 37, and 49 shows that this object should be grouped in family 2 concept. The result of this process has reduced the 9 initial family concepts down to three with family 1 concept corresponding to the collagen family, family 37 concept
corresponding to the myotubularin family and family 49 concept corresponding to the receptor family.

This same data set is also processed using the proteins as the objects for the concept lattices. Figures 7-17-a, 7-17-b, and 7-17-c show the concept lattices for the three individual families and then 7-18 shows the combined concept lattice for all three families. Looking at these figures, one sees that for the microtubulin and collagen families, the concept lattices are identical but for the receptor family there are some differences. For both collagen and microtubulin families, all proteins for the same genes have exactly the same set of GO terms, but it is not the case for the receptor family. For example, the gene FGFR1 has the protein AAA35838 which has fewer annotations than the other proteins associated with FGFR1; FGFR2 has the protein BAC45037 which has fewer annotations than the other proteins associated with FGFR2. Several other genes in receptor family also have this same situation so that the concept lattice for the receptor family is different when the objects selected for the concept lattice are proteins instead of genes. Getting the different concept lattice results from doing the FCA on both kinds of objects may indicate that the annotation process should be reviewed for the receptor family.

![Figure 7-17-a Concept Lattice for Myotubularin Family using Protein Objects](image.png)
Figure 7-17-b  Concept Lattice for Receptor Family using Protein Objects

Figure 7-17-c  Concept Lattice for Collagen Family using Protein Objects
7.4 Concept Similarity

To further explore the use of the concept lattice software, the breast cancer data from UCCHMC is used to evaluate both the concept and concept lattice similarity function. For this data, the objective is to discover which genes in the test set are most strongly related to the cancer causing set of genes. Here the concept similarity functionality is also used but concept similarities are calculated based on only the intents of two concept lattices since the extents for the training and test sets are based on different sets of genes; therefore the parameter w in the concept similarity formula is set to 0.
First of all, based on the training concept lattice and the test concept lattice, we need to find out all pairs of concepts that have high similarity by setting the threshold value to 0.9. The list of all concept pairs we found is attached in appendix.

Among all those highly similar concept pairs, we need to rank all the genes included in those concepts and compare the rank result with the prioritization result produced by UCCHMC. In order to rank all those found genes, we need a function to evaluate its potential as a breast cancer causing gene. 

\[ P_g = \sum \text{Concept Similarity} \times \text{Num GO terms} \]

where \( P_g \) is the potential of being a breast cancer causing gene. Here the concept similarity is the similarity of the concept pair where the test concept includes gene \( g \). The number of GO terms in the test concept is taken into account because the more GO terms the concept has, the more specific it is. We should give some extra score to reward the specificity of a concept. Finally, we get the rank list of all potential breast cancer causing genes in the test set:
Table 7-4 Rank of Potential Breast Cancer Causing Genes Found in the Test Set

<table>
<thead>
<tr>
<th>Rank</th>
<th>Gene Symbol</th>
<th>Potential Score</th>
<th>Rank in UCCHMC’s Prioritization Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>MKI67 [4288]</td>
<td>7.829537983</td>
<td>28</td>
</tr>
<tr>
<td>2</td>
<td>ZRANB1 [54761]</td>
<td>5.815370679</td>
<td>56</td>
</tr>
<tr>
<td>3</td>
<td>BCCIP [56647]</td>
<td>4.829537983</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>EMX2 [2018]</td>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td>5</td>
<td>BUB3 [9184]</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>TACC2 [10579]</td>
<td>4</td>
<td>57</td>
</tr>
<tr>
<td>7</td>
<td>DGCK1 [1793]</td>
<td>2.815370679</td>
<td>21</td>
</tr>
<tr>
<td>8</td>
<td>EIF3S10 [3661]</td>
<td>2.815370679</td>
<td>16</td>
</tr>
<tr>
<td>9</td>
<td>SEC23IP [11196]</td>
<td>2.815370679</td>
<td>15</td>
</tr>
<tr>
<td>10</td>
<td>ATEL [11101]</td>
<td>2</td>
<td>11</td>
</tr>
<tr>
<td>11</td>
<td>FGFR2 [2263]</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>12</td>
<td>OAT [4942]</td>
<td>2</td>
<td>43</td>
</tr>
<tr>
<td>13</td>
<td>CTBP2 [1488]</td>
<td>1</td>
<td>47</td>
</tr>
<tr>
<td>14</td>
<td>HMX2 [3167]</td>
<td>1</td>
<td>20</td>
</tr>
<tr>
<td>15</td>
<td>VAX1 [11023]</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>16</td>
<td>C10orf137 [26098]</td>
<td>1</td>
<td>9</td>
</tr>
<tr>
<td>17</td>
<td>PLEKHA1 [59338]</td>
<td>1</td>
<td>52</td>
</tr>
<tr>
<td>18</td>
<td>GRK5 [2369]</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>19</td>
<td>GALNAC4S-6ST [51363]</td>
<td>1</td>
<td>14</td>
</tr>
<tr>
<td>20</td>
<td>C10orf89 [118672]</td>
<td>1</td>
<td>76</td>
</tr>
<tr>
<td>21</td>
<td>ABUIM1 [3963]</td>
<td>1</td>
<td>22</td>
</tr>
<tr>
<td>22</td>
<td>HTRA1 [5654]</td>
<td>1</td>
<td>13</td>
</tr>
<tr>
<td>23</td>
<td>RGS10 [6001]</td>
<td>1</td>
<td>10</td>
</tr>
<tr>
<td>24</td>
<td>ADAM12 [8038]</td>
<td>1</td>
<td>38</td>
</tr>
<tr>
<td>25</td>
<td>BAG3 [9524]</td>
<td>1</td>
<td>41</td>
</tr>
<tr>
<td>26</td>
<td>KIAA0157 [23172]</td>
<td>1</td>
<td>34</td>
</tr>
<tr>
<td>27</td>
<td>CUZD1 [56624]</td>
<td>1</td>
<td>40</td>
</tr>
<tr>
<td>28</td>
<td>BTBD16 [118663]</td>
<td>1</td>
<td>30</td>
</tr>
<tr>
<td>29</td>
<td>PDZD8 [118987]</td>
<td>1</td>
<td>25</td>
</tr>
</tbody>
</table>

Out of the above 29 potential breast cancer causing genes we found in the test set, there are 18 genes are also ranked in the top 29 in the prioritization result provided by UCCHMC. Among the other 11 genes, 3 of them are ranked between 30 and 39, and 4 of them are ranked between 40
and 49. However, there are still 4 of them are ranked beyond top 50 in the prioritization result. Therefore, if we just compare the top 29 in our result and that in the prioritization result, the matching accuracy is 18 / 29, or 62.07%. The accuracy could be 72.41% or 86.21% if we extend the list in the prioritization result to top 39 or top 49.

Another example is to compare similarity between pairs of proteins. The data is provided by Catia Pesquita, a PhD Student at University of Lisbon. She has used this data set in her tool to evaluate semantic similarity measures used for biological data (Pesquita et al, 2008). There are totally three files provided. One is “highsim” file containing 264 pairs of proteins and the similarity value for each pair which are generally high (greater than 0.9). Another file is called “lowsim”, containing 379 pairs of proteins and the similarity value for each pair which are generally low (less than 0.1). All the similarity values in these two files are calculated using Pesquita’s own similarity measure “GIC”. The third given file contains all the associated GO terms of every protein in highsim and lowsim files. A concept lattice can be created using proteins as objects and GO terms as attributes. The method to compare a pair of proteins in this paper is to do concept similarity between the concepts associated with the two objects. Each object is associated with exactly one concept and it appears that the associated concept is the one which has all the associated attributes as its intent. To calculate concept similarity, C-SS-Tversky (discussed in section 5.4, using Lin’s measure for GO similarities) measure is used. Table 7-5 shows a statistics and comparison of the similarity results for both highsim and lowsim files using GIC and C-SS-Tversky measures.

<table>
<thead>
<tr>
<th></th>
<th>Highsim</th>
<th>Lowsim</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>average</td>
<td>standard dev.</td>
</tr>
<tr>
<td>GIC</td>
<td>0.959953682</td>
<td>0.0315238</td>
</tr>
<tr>
<td>C-SS-Tversky</td>
<td>0.967990788</td>
<td>0.0558651</td>
</tr>
</tbody>
</table>

Table 7-5 Comparison of the Similarity Results using GIC and C-SS-Tversky
The table shows that the two measures did very similar results on comparing protein pairs of highsim file. A Kendall Tau test was run on two sets of similarity measures for the highsim file. The resulting correlation coefficient is 0.707 which demonstrates the FCA method correlates with the GIC approach. For protein pairs in lowsim file, the average similarity value is somewhat different indicating that the some pairs still share a few GOs between two proteins which increases their intent similarity. The correlation coefficient is 0.346 between the two similarities is very poor. However, an average similarity value of 0.278116 for the C-SS-Tversky measures still indicates that protein pairs in lowsim file have low similarities. One difference between the two approaches to assessing similarity is that the FCA method is using an information content measure based solely on the structure of the GO while the GIC uses an information content measure based on a corpus of annotation files and make reflect more accurately the significance of each annotation than simply using the GO structure. These results need to be more thoroughly investigated.

### 7.5 Concept Approximation

In order to better understand the various approaches to concept approximation found in the research literature, this thesis investigated their relationships. First a simple formal context example presented in (Saquer and Deogun, 2001) is used to compare the results of the various approaches. Table 7-6 shows the example formal context, and figure 7-19 shows the visualization of the corresponding concept lattice.
Table 7-6  A formal context example in Saquer’s paper

<table>
<thead>
<tr>
<th></th>
<th>α</th>
<th>b</th>
<th>c</th>
<th>d</th>
<th>e</th>
<th>f</th>
<th>h</th>
<th>i</th>
<th>j</th>
<th>k</th>
<th>l</th>
<th>x</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td></td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 7-19  Visualization of Concept Lattice of Table 7-6
Figure 7-20  Extent and Intent of Each Concept in Figure 7-19

The following table 7-7 shows five object set A and the results of each set using the different concept approximation methods. Results show that the lower and upper concept approximation of five object set A are the same for Hu’s and Yao’s approaches, but results for Saquer’s approach are different for set A2, A3 and A5. Take set A2 as an example. In Hu’s approach, we can easily find out that concepts 1, 3, 10, 14, 16, 17, 19 and 21 are join-irreducible concepts since there is only one child concept for each of them, but none of their extents is a subset of set A2 ({2, 3}). Therefore, the low approximation is an empty set using Hu’s approach. In Yao’s approach, because there is no other concept of which the extent is a subset of A2 except the bottom node, the low approximation will be the extent of the bottom node which is an empty set. In Saquer’s approach, the first step is to find out the equivalence classes of the formal concept, which is the set \{\{1\}, \{2\}, \{3, 11\}, \{4\}, \{5, 12, 13\}, \{6\}, \{7\}, \{8\}, \{9\}, \{10\}\}. Then we list all subsets of A2, which should be \{\}, \{2\}, \{3\}, \{2, 3\}\}. Subset \{2\} exists in the equivalence classes, so LA(A) should be \{2\}. Different from Hu’s and Yao’s approaches, the extent of the
low concept approximation is not just the low approximation object set. So next, we find out the set $\text{LA}(A)'$ which contains all associated attributes with object 2: \{d, f, h\}. Finally, the extent of the low concept approximation result is $\text{LA}(A)' = \{2, 4, 5, 8, 12, 13\}$. Instead of giving an empty set, the Saquer’s approach produces the above set which can help to decide a more accurate approximation concept that can best describe the given set $A_2$. It is the same case for set $A_3$ or $A_5$. Thus, the Saquer’s approach produces a better lower approximation results than Hu’s and Yao’s approaches.

<table>
<thead>
<tr>
<th></th>
<th>$A_1={3,4,5}$</th>
<th>$A_2={2,3}$</th>
<th>$A_3={4,9}$</th>
<th>$A_4={4,5,6}$</th>
<th>$A_5={5,6,8}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hu</td>
<td>$\text{LA}(A)'$</td>
<td>{}</td>
<td>${4}$</td>
<td>${4,8}$</td>
<td>${8}$</td>
</tr>
<tr>
<td></td>
<td>$\text{UA}(A)'$</td>
<td>${2,3,4,5,6,7,8,11,12,13}$</td>
<td>${2,3,4,5,6,7,8,1}$</td>
<td>${4,9,10}$</td>
<td>${4,5,6,8,12,13}$</td>
</tr>
<tr>
<td>Yao</td>
<td>$\text{LA}(A)'$</td>
<td>{}</td>
<td>${4}$</td>
<td>${4,8}$</td>
<td>${8}$</td>
</tr>
<tr>
<td></td>
<td>$\text{UA}(A)'$</td>
<td>${2,3,4,5,6,7,8,11,12,13}$</td>
<td>${2,3,4,5,6,7,8,1}$</td>
<td>${4,9,10}$</td>
<td>${4,5,6,8,12,13}$</td>
</tr>
<tr>
<td>Saquer</td>
<td>$\text{LA}(A)'$</td>
<td>{}</td>
<td>${4}$</td>
<td>${4,8}$</td>
<td>${8}$</td>
</tr>
<tr>
<td></td>
<td>$\text{UA}(A)'$</td>
<td>${2,3,4,5,6,7,8,11,12,13}$</td>
<td>${2,3,4,5,6,7,8,1}$</td>
<td>${4,9,10}$</td>
<td>${4,5,6,8,12,13}$</td>
</tr>
</tbody>
</table>

**Table 7-7 Result of Extents of LA and UA Concepts of Object Sets**

The real-world cancer data set is used to further examine the concept approximation query in order to see if it might help identify genes from the test set that are highly associated with those in the training set. The upper and lower approximations for a set of attributes are queried for within the combined concept lattice. The training set contains 15 cancer-causing genes and 167 associated GO terms in total. When we only display the GO terms that are associated with certain numbers of genes or more, i.e. at least 5 genes, we get the results:

GO_0005634: [ATM, BARD1, BRCA1, BRCA2, PALB2, PIK3CA, PPM1D, RAD51, RB1CC1, SLC22A18, TP53]
GO_0007049: [ATM, BRCA1, PIK3CA, PPM1D, RAD51, RB1CC1]
GO_0005515: [BARD1, BRCA1, BRCA2, BRIP1, CASP8, PALB2, PIK3CA, RAD51, RB1CC1, TP53]
The set of these three GO terms (GO_0005634, GO_0007049, GO_0005515) contains the most important terms among the total 167 ones because for each GO term at least 5 out of 15 cancer-causing genes are associated to it. Here we use Saquer’s approach to calculate the UA and LA concepts for this set in the concept lattice of the test set.

Results show that concept (\{MKI67\}, \{GO_0005634, GO_0007049, GO_0005515, GO_0000166, GO_0005524, GO_0000074, GO_0005730, GO_0008283\}) is the UA and LA concept and best describes the set (GO_0005634, GO_0007049, GO_0005515) from training set. Therefore, in the extent of this concept, genes MKI67 from test set are considered highly associated with genes from training set and thus cancer causing as well.

8 Conclusions and Future Work

The objective of this thesis is expanding the functionality of the original QUOTA system (Sun 2007) which provides querying capabilities on a set of objects that are annotated with terms coming from an ontological terminology. This thesis has added new view on this same kind of data, i.e., the set of objects and their associated annotations which serve as attributes of the objects are interpreted as a formal context. The significance of this thesis is its development of a sophisticated yet easy to use FCA software tool that provides the user with 3D visualization of results. This FCA software tool incorporates the structure of the ontology from which attributes are taken to allow granulation of the concept lattice from coarse to finer and vice versa. Many of the proposed algorithms in the research literature that operate on concept lattices such a concept approximation and concept similarity have been implemented. In addition many useful querying capabilities on the structure of a concept lattice are included. This thesis has demonstrated the new functionality using the problem domain of bioinformatics where the formal context consists of gene or protein objects and annotation terms from the Gene Ontology as the objects’ attributes. This problem domain is an important one where the use of FCA has recently begun to intensify.
The concept lattices created using this software can be stored in multiple forms, a text output file which can serve as input to the system for further querying upon and as a gml file which can be input to VTK to produce a 3D visualization of a concept lattice.

The development of the new granulation algorithm operates on the original concept lattice to modify it instead of rebuilding it from a new formal context using only the attributes satisfying the selection criteria and then creating a whole new concept lattice. The complexity analysis of this algorithm has shown that for lattices with a large number of concepts this approach is more efficient. The granulated concept lattice helps the user better understand the overall or high level view of the formal context. The granulation feature allows three different methods of specifying the selection criteria for annotations. This thesis not only developed the new functionality allow creation and visualization of a variety of concept lattices, but also added sophisticated querying capabilities that allow the user to explore the concepts in and the structure of the created concept lattices and to compare two concept lattices for similarity. This additional querying capability has been demonstrated on several examples and on real world data sets to illustrate its usefulness in exploring relationships among the data and discovering knowledge existing in this data.

This thesis has added to QUOTA the very useful functionality to build and query concept lattices using objects and their associated annotations taken from an ontological terminology, but other improvements are needed to further enhance the value of this FCA software. Examples of such enhancements include:

1) Additional visualization features for concept lattices that would allow storing the intents and extents of the concepts in the gml file so that mousing over the concept nodes in VTK could reveal these would be very valuable for better integrating the content of the concept lattice with its visualization. Currently, only the node ID for each concept is shown in the VTK visualization.

2) An algorithm to read in a concept lattice and directly create its OOCL and POCL without knowing and building its formal context could be interesting future research. Additional
research should investigate the usefulness of these two alternative views on a formal context for discovering interesting relationships.

3) More work should continue on the granulation algorithm to further investigate ways to decrease its complexity. As part of this work, filtering annotations based on evidence codes associated with the annotation should be added as new criteria.

4) A detailed algorithm and implementation needs to be developed based on the simple example of finding the families in the GPD194 data in order to locate strong clusters of objects and compare it to other complex clustering techniques.

5) Research should investigate the development of parallel algorithms for concept lattice construction in order to shorten the running time on creating large concept lattices.

This thesis has developed a valuable FCA software tool that can be used to explore bioinformatics data. Much more research on the application of this software in this problem domain needs to be undertaken to thoroughly explore its usefulness and to enhance its capabilities further to provide the bioinformatics researchers with tools to assist them in discovering new knowledge.
9 References


Sun, Y. (2007). Querying with Terminological Ontologies and Their Annotations, Master of Computer Science, Miami University, Oxford, OH.


Appendix

Appendix: Concept Information of Figure 7-9, 7-10, and 7-11 (Training Data of Breast Cancer Example)

Figure 7-9:
0: []; [472, 580, 672, 675, 841, 3845, 5002, 5290, 5888, 7157, 8493, 9821, 11200, 79728, 83990]  
1: [GO_0003677, GO_0005634]; [472, 672, 83990]  
2: [GO_0004674, GO_0005634, GO_0007049, GO_0016740]; [472, 11200]  
3: [GO_0004674, GO_0003677, GO_0005634, GO_0006281, GO_0007049, GO_0007094, GO_0007131, GO_0007165, GO_0010212, GO_0016303, GO_0016740, GO_0016773, GO_0045786, GO_0047485]; [472]  
4: [GO_0005634]; [472, 580, 672, 675, 5888, 7157, 8493, 9821, 11200, 79728, 83990]  
5: [GO_0006281, GO_0005634]; [472, 672, 675, 5888]  
6: [GO_0006281, GO_0003677, GO_0005634, GO_0007049, GO_0045786]; [472, 672]  
7: [GO_0007049, GO_0005634]; [472, 672, 7157, 8493, 9821, 11200]  
8: [GO_0007131, GO_0005634, GO_0006281]; [472, 5888]  
9: [GO_0007165, GO_0016303, GO_0016740]; [472, 5290]  
10: [GO_0016740]; [472, 5290, 11200]  
11: [GO_0045786, GO_0005634, GO_0007049]; [472, 672, 7157, 9821]  
12: [GO_0047485, GO_0005634, GO_0007049, GO_0045786]; [472, 7157]  
13: [GO_0000151, GO_0005634, GO_0004842, GO_0005515, GO_000622, GO_0008270, GO_0016567, GO_0031436, GO_0046872]; [580, 672]  
14: [GO_0000151, GO_0003677, GO_0005634, GO_0006281, GO_0007049, GO_0045786, GO_004842, GO_0005515, GO_0005622, GO_0008270, GO_0016567, GO_0031436, GO_0046872, GO_0046872, GO_0000075, GO_0000072, GO_00000074, GO_0003674, GO_0003713, GO_0005575, GO_0006301, GO_0006357, GO_0006359, GO_0006633, GO_0006978, GO_0007059, GO_0008274, GO_0008630, GO_0015631, GO_0016481, GO_0019900, GO_0031436, GO_0031441, GO_0042325, GO_0042803, GO_0043065, GO_0043066, GO_0045732, GO_0045726, GO_0045926, GO_0045982, GO_0000075, GO_0000724, GO_0003674, GO_0003713, GO_0005575, GO_0006301, GO_0006357, GO_0006359, GO_0006633, GO_0006978, GO_0007059, GO_0008274, GO_0008630, GO_0015631, GO_0016481, GO_0019900, GO_0030521, GO_0031398, GO_0044600, GO_0050681]; [672]  
15: [GO_0000151, GO_0003677, GO_0005634, GO_0004674, GO_0007049, GO_0016740, GO_0006281, GO_0007094, GO_0007131, GO_0007165, GO_0010212, GO_0016303, GO_0016773, GO_0045786, GO_0047485, GO_0001894, GO_0004842, GO_0005515, GO_0005622, GO_0005737, GO_0006974, GO_0007050, GO_0008270, GO_0016567, GO_0019900, GO_0031436, GO_0031441, GO_0042325, GO_0042803, GO_0043065, GO_0043066, GO_0045732, GO_0045726, GO_0045926, GO_0045982, GO_0000075, GO_0000724, GO_0003674, GO_0003713, GO_0005575, GO_0006301, GO_0006357, GO_0006359, GO_0006633, GO_0006978, GO_0007059, GO_0008274, GO_0008630, GO_0015631, GO_0016481, GO_0019900, GO_0030521, GO_0031398,
| GO_0042127, GO_0042981, GO_0045717, GO_0045739, GO_0045893, GO_0046600, GO_0050681, GO_0000074, GO_0003697, GO_0004402, GO_0005615, GO_0006325, GO_0006338, GO_0007090, GO_0007093, GO_0016563, GO_0030141, GO_0045449, GO_0042149, GO_0042771, GO_0046902, GO_0051097, GO_0051262, GO_0005524, GO_0000739, GO_0003700, GO_0004518, GO_0005507, GO_0005524, GO_0000794, GO_0003684, GO_0003690, GO_0005524, GO_0006268, GO_0006312, GO_0007126, GO_0008094, GO_0016887, GO_0017111, GO_0043142, GO_0043565, GO_0051106, GO_0051260, GO_0000739, GO_0003700, GO_0004518, GO_0005507, GO_0005626, GO_0005654, GO_0006281, GO_0000724 | \[\] |
| 16 | \[GO_0005515, GO_0005634, GO_0000151, GO_0003723, GO_0004842, GO_0005515, GO_0005622, GO_0005737, GO_0006974, GO_0007050, GO_0008270, GO_0016567, GO_0019900, GO_0031436, GO_0031441, GO_0042325, GO_0042803, GO_0043065, GO_0043066, GO_0045732, GO_0046826, GO_0046872, GO_0046982\]; \[580\] |
| 17 | \[GO_0005515\]; \[580, 672, 675, 841, 3845, 5888, 7157, 9821, 11200, 83990\] |
| 18 | \[GO_0005515, GO_0003677, GO_0005634, GO_0006357\]; \[672, 83990\] |
| 19 | \[GO_0005515, GO_0004674, GO_0005634, GO_0007049, GO_0016740, GO_0008630, GO_0000166, GO_0006468, GO_0005524, GO_0000287, GO_0000077, GO_0016605\]; \[11200\] |
| 20 | \[GO_0005515, GO_0005634\]; \[580, 672, 675, 5888, 7157, 9821, 11200, 83990\] |
| 21 | \[GO_0005515, GO_0005634, GO_0006281, GO_0000724\]; \[672, 675, 5888\] |
| 22 | \[GO_0005515, GO_0005634, GO_00007049\]; \[672, 7157, 9821, 11200\] |
| 23 | \[GO_0005515, GO_0005634, GO_0006281, GO_0007131, GO_0005622, GO_0000724, GO_0003697, GO_0042802, GO_0000166, GO_0000150, GO_0000794, GO_0003684, GO_0003690, GO_0005524, GO_0000628, GO_00006312, GO_0007126, GO_0008094, GO_0016887, GO_0017111, GO_0043142, GO_0043565, GO_0016605, GO_0004003, GO_0006302, GO_0000077, GO_0016605, GO_0004003, GO_0006302, GO_0016818\]; \[\] |
| 24 | \[GO_0005515, GO_0005634, GO_0006281, GO_0000724\]; \[672, 7157, 9821\] |
| 25 | \[GO_0005515, GO_0005634, GO_00007049, GO_00045786\]; \[672, 7157, 9821\] |
Figure 7-10:

0: []; [472, 580, 672, 675, 841, 3845, 5002, 5290, 5888, 7157, 8493, 9821, 11200, 79728, 83990]
1: [GO_0003677, GO_0006974, GO_0000074, GO_0043231, GO_0006259]; [472, 672, 83990]
2: [GO_0007049, GO_0043231]; [472, 672, 7157, 8493, 9821, 11200]
3: [GO_0007049, GO_0003677, GO_0045786, GO_0006974, GO_0000074, GO_0043231, GO_0006259]; [472, 672]
4: [GO_0007165, GO_0016740, GO_0004428, GO_0001727]; [472, 5290]
5: [GO_0007165, GO_0003677, GO_0007049, GO_0010212, GO_0016740, GO_0016773, GO_0045786, GO_0047485, GO_0006974, GO_0000074, GO_0004428, GO_0043231, GO_0004672, GO_00006259, GO_0000087, GO_0051327, GO_0001727]; [472]
6: [GO_0016740]; [472, 5290, 11200]
7: [GO_0016740, GO_0007049, GO_0000074, GO_0004672, GO_0043231]; [472, 11200]
8: [GO_0045786, GO_0043231]; [472, 580, 672, 7157, 9821]
9: [GO_0045786, GO_0007049, GO_0043231]; [472, 672, 7157, 9821]
10: [GO_0047485, GO_0007049, GO_0045786, GO_0006974, GO_0043231, GO_0006259]; [472, 7157]
11: [GO_0000151, GO_0045786, GO_0005515, GO_0005622, GO_0006974, GO_0008270, GO_0046872, GO_0042981, GO_0043231, GO_0019787, GO_0044267, GO_0043412, GO_00000152, GO_0016070, GO_0019219, GO_0031324, GO_0043068]; [580, 672]
12: [GO_0000151, GO_0003677, GO_0007049, GO_0045786, GO_0005515, GO_0005622, GO_0006974, GO_0008270, GO_0046872, GO_0003674, GO_0003713, GO_0005575, GO_00007059, GO_0015631, GO_0019899, GO_0042127, GO_0042981, GO_0000074, GO_0006350, GO_0043231, GO_0006259, GO_0019787, GO_0044267, GO_0043412, GO_00000152, GO_0016070, GO_0019219, GO_0031324, GO_0043068, GO_0008610, GO_0016053, GO_0019752, GO_0044255, GO_0042772, GO_00030330, GO_0043234, GO_0044446, GO_0044444, GO_0043232, GO_0042770, GO_00030522, GO_0019217, GO_0045833, GO_0019216, GO_0031326, GO_0009890, GO_0031325, GO_0046605, GO_0051298, GO_0035257]; [672]
13: [GO_0000151, GO_0003677, GO_0007049, GO_0045786, GO_0007165, GO_0016740, GO_0010212, GO_0016773, GO_0047485, GO_0001894, GO_0003723, GO_0005515, GO_0005622, GO_0005737, GO_0006974, GO_0008270, GO_0019900, GO_0042803, GO_0046872, GO_0046982, GO_0003674, GO_0003713, GO_0005575, GO_0007059,
Figure 7-11:
0: [GO_0043227, GO_0043226, GO_0044464]; [472, 580, 672, 675, 841, 3845, 5002, 5290, 5888, 7157, 8493, 9821, 11200, 79728, 83990]
1: [GO_0016740, GO_0043227, GO_0043226, GO_0044464, GO_0005515, GO_0009719, GO_0006950, GO_0008152, GO_0050875, GO_0050794, GO_0050791]; [472, 675, 5290, 11200]
2: [GO_0005515, GO_0043227, GO_0043226, GO_0044464, GO_0009719, GO_0006950, GO_0008152, GO_0050875, GO_0050794, GO_0050791]; [472, 580, 672, 675, 841, 3845, 5002, 5290, 5888, 7157, 9821, 11200, 79728, 83990]
3: [GO_0003674, GO_0043227, GO_0043226, GO_0044464, GO_0005515, GO_0005575, GO_0016563, GO_0003676, GO_0009719, GO_0006950, GO_0008152, GO_0050875, GO_0050794, GO_0050791, GO_0007154, GO_0048519, GO_0043234, GO_0016874, GO_0043167, GO_0044422, GO_0044421, GO_0031982, GO_0051869, GO_0015075, GO_0005386, GO_0015238, GO_0042221, GO_0040008, GO_0035264, GO_0043233, GO_0009607, GO_0007568, GO_0050790, GO_0040007, GO_0048856, GO_0051716, GO_0009605, GO_0004386]; []
4: [GO_0003674, GO_0043227, GO_0043226, GO_0044464, GO_00016740, GO_0005515, GO_0005575, GO_0005575, GO_0016563, GO_0003676, GO_0009719, GO_0006950, GO_0008152, GO_0050875, GO_0050794, GO_0050791, GO_0007154, GO_0048519, GO_0043234, GO_0016874, GO_0043167, GO_0044422, GO_0044421, GO_0031982, GO_0051869, GO_0015075, GO_0005386, GO_0015238, GO_0042221, GO_0040008, GO_0035264, GO_0043233, GO_0009607, GO_0007568, GO_0050790, GO_0040007, GO_0048856, GO_0051716, GO_0009605, GO_0004386]; []
5: [GO_0016563, GO_0043227, GO_0043226, GO_0044464, GO_0005515, GO_0003676, GO_0009719, GO_0006950, GO_0008152, GO_0050875, GO_0050794, GO_0050791]; [672,
675] 6: [GO_0016563, GO_0043227, GO_0043226, GO_0044464, GO_0016740, GO_0005515, GO_0003676, GO_0009719, GO_0006950, GO_0008152, GO_0050875, GO_0050794, GO_0050791, GO_0044421, GO_0031982]; [675] 7: [GO_0004871, GO_0043227, GO_0043226, GO_0044464, GO_0005515, GO_0016787, GO_0009719, GO_0006950, GO_0008152, GO_0050875, GO_0050794, GO_0050791, GO_0007154, GO_0048518, GO_0016265, GO_0043228]; [841] 8: [GO_0000166, GO_0043227, GO_0043226, GO_0044464, GO_0005515, GO_0009719, GO_0006950, GO_0008152, GO_0050875, GO_0050794, GO_0050791]; [3845, 5888, 7157, 11200, 83990] 9: [GO_0000166, GO_0043227, GO_0043226, GO_0044464, GO_0005515, GO_0009719, GO_0006950, GO_0008152, GO_0050875, GO_0050794, GO_0050791, GO_0007154, GO_0048518, GO_0016265, GO_0043228]; [5002] 11: [GO_0000166, GO_0043227, GO_0043226, GO_0044464, GO_0005515, GO_0000166, GO_0016787, GO_0003676, GO_0009719, GO_0006950, GO_0008152, GO_0050875, GO_0050794, GO_0050791, GO_0044422, GO_0048518, GO_0016265, GO_0043228]; [5888] 12: [GO_0000166, GO_0043227, GO_0043226, GO_0044464, GO_0005515, GO_0000166, GO_0016787, GO_0003676, GO_0009719, GO_0006950, GO_0008152, GO_0050875, GO_0050794, GO_0050791, GO_0007154, GO_0048518, GO_0016265, GO_0043228, GO_0040008, GO_0035264, GO_0043233, GO_0009607, GO_0007568, GO_0050790, GO_0040007, GO_0048856, GO_0051716, GO_0009605]; [7157] 13: [GO_00016787, GO_0043227, GO_0043226, GO_0044464, GO_0009719, GO_0006950, GO_0008152, GO_0050875, GO_0050794, GO_0050791]; [841, 3845, 5888, 7157, 8493, 83990] 14: [GO_0016787, GO_0043227, GO_0043226, GO_0044464, GO_0005515, GO_0009719, GO_0006950, GO_0008152, GO_0050875, GO_0050794, GO_0050791]; [3845, 841, 5888, 7157, 83990] 15: [GO_0016787, GO_0043227, GO_0043226, GO_0044464, GO_0005515, GO_0000166, GO_0009719, GO_0006950, GO_0008152, GO_0050875, GO_0050794, GO_0050791]; [3845, 5888, 7157, 83990] 16: [GO_0003676, GO_0043227, GO_0043226, GO_0044464, GO_0005515, GO_0009719, GO_0006950, GO_0008152, GO_0050875, GO_0050794, GO_0050791]; [472, 580, 672, 675, 5888, 7157, 83990] 17: [GO_0003676, GO_0043227, GO_0043226, GO_0044464, GO_0016740, GO_0005515, GO_0009719, GO_0006950, GO_0008152, GO_0050875, GO_0050794, GO_0050791]; [472, 675] 18: [GO_0003676, GO_0043227, GO_0043226, GO_0044464, GO_0005515, GO_0000166, GO_0016787, GO_0009719, GO_0006950, GO_0008152, GO_0050875, GO_0050794,
GO_0050791]; [5888, 7157, 83990]
19: [GO_0009719, GO_0043227, GO_0043226, GO_0044464, GO_0006950, GO_0008152, GO_0050875]; [472, 580, 672, 675, 841, 3845, 5002, 5290, 5888, 7157, 8493, 9821, 11200, 83990]
20: [GO_0050794, GO_0043227, GO_0043226, GO_0044464, GO_0009719, GO_0006950, GO_0008152, GO_0050875, GO_0050791]; [472, 580, 672, 675, 841, 3845, 5290, 5888, 7157, 8493, 9821, 11200, 83990]
21: [GO_0007154, GO_0043227, GO_0043226, GO_0044464, GO_0005515, GO_0009719, GO_0006950, GO_0008152, GO_0050875, GO_0050794, GO_0050791]; [472, 672, 841, 3845, 5290, 7157]
22: [GO_0007154, GO_0043227, GO_0043226, GO_0044464, GO_0005515, GO_0009719, GO_0006950, GO_0008152, GO_0050875, GO_0050794, GO_0050791, GO_0048519]; [472, 5290]
23: [GO_0007154, GO_0043227, GO_0043226, GO_0044464, GO_0005515, GO_0009719, GO_0006950, GO_0008152, GO_0050875, GO_0050794, GO_0000166, GO_0016787]; [3845, 7157]
24: [GO_0007154, GO_0043227, GO_0043226, GO_0044464, GO_0005515, GO_0009719, GO_0006950, GO_0008152, GO_0050875, GO_0050794, GO_00016787]; [841, 3845, 7157]
25: [GO_0007154, GO_0043227, GO_0043226, GO_0044464, GO_0005515, GO_0009719, GO_0006950, GO_0008152, GO_0050875, GO_0050794, GO_0003676, GO_0048519]; [472, 7157]
26: [GO_0007154, GO_0043227, GO_0043226, GO_0044464, GO_0016740, GO_0005515, GO_0009719, GO_0006950, GO_0008152, GO_0050875, GO_0050794, GO_0003676, GO_0009628, GO_0048519]; [472]
27: [GO_0009628, GO_0043227, GO_0043226, GO_0044464, GO_0009719, GO_0006950, GO_0008152, GO_0050875, GO_0050794, GO_0050791, GO_0003676, GO_0009628, GO_0048519]; [472]
28: [GO_0009628, GO_0043227, GO_0043226, GO_0044464, GO_0016787, GO_0009719, GO_0006950, GO_0008152, GO_0050875, GO_0050794, GO_00016787, GO_0048519, GO_0003676, GO_0000166, GO_0048519]; [5290, 7157]
29: [GO_0009628, GO_0043227, GO_0043226, GO_0044464, GO_0016787, GO_0009719, GO_0006950, GO_0008152, GO_0050875, GO_0050794, GO_0050791, GO_0048519]; [5290, 7157]
30: [GO_0009628, GO_0043227, GO_0043226, GO_0044464, GO_0016740, GO_0005515, GO_0009719, GO_0006950, GO_0008152, GO_0050875, GO_0050794, GO_0000166, GO_0048519]; [5290, 7157]
31: [GO_0009628, GO_0043227, GO_0043226, GO_0044464, GO_0016740, GO_0005515, GO_0009719, GO_0006950, GO_0008152, GO_0050875, GO_0050794, GO_0050791, GO_0048519, GO_0043167, GO_0009607]; [7157]
32: [GO_0009628, GO_0043227, GO_0043226, GO_0044464, GO_0016787, GO_0009719, GO_0006950, GO_0008152, GO_0050875, GO_0050794, GO_0050791, GO_0003676]; [7157]
580, 672, 7157]
33: [GO_0048519, GO_0043227, GO_0043226, GO_0044464, GO_0005515, GO_0009719, GO_0006950, GO_0008152, GO_0050875, GO_0050794, GO_0050791, GO_0007154]; [472, 672, 5290, 7157]
34: [GO_0043234, GO_0043227, GO_0043226, GO_0044464, GO_0009719, GO_0006950, GO_0008152, GO_0050875, GO_0050794, GO_0050791, GO_0007154, GO_0048519, GO_0048518, GO_0016265, GO_0040008, GO_0035264]; [580, 672, 5290]
35: [GO_0043234, GO_0043227, GO_0043226, GO_0044464, GO_0016740, GO_0005515, GO_0009719, GO_0006950, GO_0008152, GO_0050875, GO_0050794, GO_0050791, GO_0007154, GO_0008152, GO_0050875, GO_0050794, GO_0050791, GO_0007154, GO_0048519, GO_0048518, GO_0016265, GO_0040008, GO_0035264]; [580, 672, 5290]
36: [GO_0043234, GO_0043227, GO_0043226, GO_0044464, GO_0009719, GO_0006950, GO_0008152, GO_0050875, GO_0050794, GO_0050791, GO_0048519, GO_0048518, GO_0016265]; [580, 672, 5290]
37: [GO_0043234, GO_0043227, GO_0043226, GO_0044464, GO_0016740, GO_0005515, GO_0009719, GO_0006950, GO_0008152, GO_0050875, GO_0050794, GO_0050791, GO_0007154, GO_0008152, GO_0050875, GO_0050794, GO_0050791, GO_0007154, GO_0048519, GO_0048518, GO_0016265, GO_0040008, GO_0035264]; [580, 672, 5290]
38: [GO_0043234, GO_0043227, GO_0043226, GO_0044464, GO_0005515, GO_0009719, GO_0006950, GO_0008152, GO_0050875, GO_0050794, GO_0050791, GO_0048519, GO_0048518, GO_0016265]; [580, 672, 5290]
39: [GO_0043234, GO_0043227, GO_0043226, GO_0044464, GO_0005515, GO_0009719, GO_0006950, GO_0008152, GO_0050875, GO_0050794, GO_0050791, GO_0007154, GO_0008152, GO_0050875, GO_0050794, GO_0050791, GO_0007154, GO_0048519, GO_0016874, GO_0043167, GO_0044422, GO_0048518, GO_0016265]; [580, 672, 5290]
40: [GO_0043234, GO_0043227, GO_0043226, GO_0044464, GO_0005515, GO_0009719, GO_0006950, GO_0008152, GO_0050875, GO_0050794, GO_0050791, GO_0007154, GO_0048519, GO_0016874, GO_0043167, GO_0044422, GO_0048518, GO_0016265, GO_0051179]; [580, 672, 5290]
41: [GO_0043234, GO_0043227, GO_0043226, GO_0044464, GO_0005515, GO_0009719, GO_0006950, GO_0008152, GO_0050875, GO_0050794, GO_0050791, GO_0000166, GO_0016787, GO_0007154, GO_0050874, GO_0051869]; [3845]
42: [GO_0043234, GO_0043227, GO_0043226, GO_0044464, GO_0005515, GO_0009719, GO_0006950, GO_0008152, GO_0050875, GO_0050794, GO_0050791, GO_0003676, GO_0048519, GO_0016874, GO_0043167, GO_0044422, GO_0048518, GO_0016265]; [580, 672, 5290]
43: [GO_0043234, GO_0043227, GO_0043226, GO_0044464, GO_0005515, GO_0009719, GO_0006950, GO_0008152, GO_0050875, GO_0050794, GO_0050791, GO_0003676, GO_0048519, GO_0016874, GO_0043167, GO_0044422, GO_0048518, GO_0016265]; [580, 672, 5290]
44: [GO_0043234, GO_0043227, GO_0043226, GO_0044464, GO_0005515, GO_0009719, GO_0006950, GO_0008152, GO_0050875, GO_0050794, GO_0050791, GO_0000166, GO_0044422, GO_0048518, GO_0016265, GO_0043233]; [7157, 11200]
45: [GO_0043167, GO_0043227, GO_0043226, GO_0044464, GO_0005515, GO_0009719, GO_0006950, GO_0008152, GO_0050875, GO_0050794, GO_0050791, GO_0003676, GO_0048519, GO_0016874, GO_0043167, GO_0044422, GO_0048518, GO_0016265]; [580, 672, 5290]
GO_0048519, GO_0044422, GO_0048518, GO_0016265]; [580, 672, 7157]
47: [GO_0043167, GO_0043227, GO_0043226, GO_0044464, GO_0005515, GO_0009719,
GO_0006950, GO_0008152, GO_0050875, GO_0050794, GO_0050791, GO_003676,
GO_0007154, GO_0048519, GO_0044422, GO_0048518, GO_0016265, GO_0043228]; [672,
7157]
48: [GO_0043167, GO_0043227, GO_0043226, GO_0044464, GO_0009719, GO_0006950,
GO_0008152, GO_0050875, GO_0050794, GO_0050791, GO_0003676, GO_0007154,
GO_0009719, GO_0006950, GO_0008152, GO_0050875, GO_0050794, GO_0050791,
GO_0048519, GO_0044422, GO_0048518, GO_0016265, GO_0043228]; [580, 672,
7157, 8493]
49: [GO_0043167, GO_0043227, GO_0043226, GO_0044464, GO_0009719, GO_0006950,
GO_0008152, GO_0050875, GO_0050794, GO_0050791, GO_0048519, GO_0043234]; [580,
672, 8493]
50: [GO_0044422, GO_0043227, GO_0043226, GO_0044464, GO_0005515, GO_0009719,
GO_0006950, GO_0008152, GO_0050875, GO_0050794, GO_0050791, GO_0003676,
GO_0007154, GO_0048519, GO_0016265]; [580, 672, 5888, 7157, 11200]
51: [GO_0044422, GO_0043227, GO_0043226, GO_0044464, GO_0005515, GO_0009719,
GO_0006950, GO_0008152, GO_0050875, GO_0050794, GO_0050791, GO_0000166,
GO_0048518, GO_0016265]; [588, 7157, 11200]
52: [GO_0044422, GO_0043227, GO_0043226, GO_0044464, GO_0005515, GO_0009719,
GO_0006950, GO_0008152, GO_0050875, GO_0050794, GO_0050791, GO_0000166,
GO_0016787, GO_0003676, GO_0048518, GO_0016265, GO_0043228]; [588, 7157]
53: [GO_0044422, GO_0043227, GO_0043226, GO_0044464, GO_0005515, GO_0009719,
GO_0006950, GO_0008152, GO_0050875, GO_0050794, GO_0050791, GO_0003676,
GO_0048518, GO_0016265]; [580, 7157, 11200]
54: [GO_0044422, GO_0043227, GO_0043226, GO_0044464, GO_0005515, GO_0009719,
GO_0006950, GO_0008152, GO_0050875, GO_0050794, GO_0050791, GO_0003676,
GO_0048518, GO_0016265]; [580, 672, 7157, 11200]
55: [GO_0044422, GO_0043227, GO_0043226, GO_0044464, GO_0005515, GO_0009719,
GO_0006950, GO_0008152, GO_0050875, GO_0050794, GO_0050791, GO_0016787,
GO_0016265, GO_0043228]; [580, 7157, 11200]
56: [GO_0044422, GO_0043227, GO_0043226, GO_0044464, GO_0005515, GO_0009719,
GO_0006950, GO_0008152, GO_0050875, GO_0050794, GO_0050791, GO_0016787,
GO_0016265, GO_0043228]; [841, 5888, 7157]
57: [GO_0044422, GO_0043227, GO_0043226, GO_0044464, GO_0005515, GO_0009719,
GO_0006950, GO_0008152, GO_0050875, GO_0050794, GO_0050791, GO_0048519,
GO_0016265]; [672, 841, 5290, 7157]
58: [GO_0044422, GO_0043227, GO_0043226, GO_0044464, GO_0005515, GO_0009719,
GO_0006950, GO_0008152, GO_0050875, GO_0050794, GO_0050791, GO_0048519,
GO_0016265]; [580, 672, 5290, 7157]
Appendix: Concept Information of Figure 7-15-a, 7-15-b, 7-15-c, and 7-16

Figure 7-15-a:
0: [GO_0006470]; [MTMR1, MTMR2, MTMR3, MTMR4, MTMR6, MTMR7, MTMR8]
1: [GO_0004722, GO_0006470, GO_0016787, GO_0008372]; [MTMR1, MTMR2, MTMR6]
2: [GO_0004725, GO_0006470, GO_0016787]; [MTMR1, MTMR6, MTMR7]
3: [GO_0004725, GO_0006470, GO_0004722, GO_0016787, GO_0008372]; [MTMR1, MTMR6]
4: [GO_0016787, GO_0006470]; [MTMR1, MTMR2, MTMR3, MTMR6, MTMR7, MTMR8]
5: [GO_0004727, GO_0006470, GO_0016787, GO_0008138]; [MTMR2, MTMR3]
6: [GO_0004727, GO_0006470, GO_0004722, GO_0016787, GO_0008372, GO_0007517, GO_0008151, GO_0008138, GO_0016020, GO_0004721, GO_0008270, GO_0006118]; []
Figure 7-15-b:
0: [GO_0004872, GO_0005524, GO_0016740, GO_0006468]; [FGFR1, FGFR2, FGFR3, FGFR4, RET, TEK, TIE1]
1: [GO_0004713, GO_0004872, GO_0005524, GO_0016740, GO_0006468]; [FGFR1, FGFR2, FGFR3, FGFR4, RET]
2: [GO_0005007, GO_0004872, GO_0005524, GO_0016740, GO_0006468, GO_0004713]; [FGFR1, FGFR2, FGFR3, FGFR4]
3: [GO_0008201, GO_0004872, GO_0005524, GO_0016740, GO_0006468, GO_0004713, GO_0005007, GO_0016021, GO_0000165, GO_0001501, GO_0008543, GO_0005887, GO_0016049, GO_0005624]; [FGFR1]
4: [GO_0016021, GO_0004872, GO_0005524, GO_0016740, GO_0006468, GO_0004713]; [FGFR1, FGFR2, RET]
5: [GO_0016021, GO_0004872, GO_0005524, GO_0016740, GO_0006468, GO_0004713, GO_0005007]; [FGFR1, FGFR2]
6: [GO_0000165, GO_0004872, GO_0005524, GO_0016740, GO_0006468, GO_0004713, GO_0005007, GO_0001501, GO_0008543, GO_0005887, GO_0016049]; [FGFR1, FGFR3]
7: [GO_0008543, GO_0004872, GO_0005524, GO_0016740, GO_0006468, GO_0004713, GO_0005007, GO_0005887]; [FGFR1, FGFR3, FGFR4]
8: [GO_0005887, GO_0004872, GO_0005524, GO_0016740, GO_0006468]; [FGFR1, FGFR3, FGFR4, TEK, TIE1]
9: [GO_00004674, GO_0004872, GO_0005524, GO_0016740, GO_0006468, GO_0004713, GO_0005007, GO_0016021]; [FGFR2]
10: [GO_00004674, GO_0004872, GO_0005524, GO_0016740, GO_0006468, GO_0004713, GO_0005007, GO_0008201, GO_0016021, GO_0000165, GO_0001501, GO_0008543, GO_0005887, GO_0016049, GO_0005624, GO_0007259, GO_0005509, GO_0007156, GO_0007165, GO_0007497, GO_0008151, GO_0005554, GO_0000004, GO_0008372, GO_0004714, GO_0008603, GO_0005952, GO_0007166, GO_0005194, GO_0007155, GO_0016020, GO_0005198, GO_0007169, GO_0007267, GO_0007498]; []
11: [GO_00007259, GO_0004872, GO_0005524, GO_0016740, GO_0006468, GO_0004713, GO_0005007, GO_0000165, GO_0001501, GO_0008543, GO_0005887, GO_0016049]; [FGFR3]
12: [GO_00005509, GO_0004872, GO_0005524, GO_0016740, GO_0006468, GO_0004713, GO_0016021, GO_0007156, GO_0007165, GO_0007497, GO_0008151, GO_0005554,
GO_0000004, GO_0008372, GO_0004714, GO_0008603, GO_0005952, GO_0007166,
GO_0005194, GO_0007155, GO_0016620]; [RET]
13: [GO_0007165, GO_0004872, GO_0005524, GO_0016740, GO_0006468, GO_0004714];
[RET, TEK, TIE1]
14: [GO_0007165, GO_0004872, GO_0005524, GO_0016740, GO_0006468, GO_0005887,
GO_0004714, GO_0005198]; [TEK, TIE1]
15: [GO_0007169, GO_0004872, GO_0005524, GO_0016740, GO_0006468, GO_0005887,
GO_0007165, GO_0004714, GO_0005198, GO_0007267]; [TEK]
16: [GO_0007498, GO_0004872, GO_0005524, GO_0016740, GO_0006468, GO_0005887,
GO_0007165, GO_0004714, GO_00005198]; [TIE1]

Figure 7-15-c:
0: []; [COL1A2, COL21A1, COL24A1, COL27A1, COL2A1, COL3A1, COL4A1, COL4A2,
COL4A3, COL4A6, COL5A3, COL9A1, COL9A2]
1: [GO_0005201]; [COL1A2, COL24A1, COL27A1, COL2A1, COL3A1, COL4A1, COL4A2,
COL4A3, COL4A6, COL5A3, COL9A1, COL9A2]
2: [GO_0005581, GO_0005201]; [COL1A2, COL2A1, COL3A1, COL5A3]
3: [GO_0008147, GO_0005201, GO_0005581, GO_0001501, GO_0005584]; [COL2A1]
4: [GO_0001501, GO_0005201]; [COL1A2, COL2A1, COL9A2]
5: [GO_0005198, GO_0007155]; [COL2A1]
6: [GO_0005198, GO_0005201, GO_0005581, GO_0008147, GO_0001501, GO_0005584,
GO_0007155, GO_0008544, GO_0007605, GO_0005585, GO_0007397, GO_0008015,
GO_0005586, GO_0005587, GO_00030198, GO_0005194, GO_0008285, GO_0009405,
GO_0007517, GO_0005588, GO_0000004, GO_0005578, GO_0005594]; []
7: [GO_0007155]; [COL21A1, COL24A1, COL4A3, COL4A6, COL5A3, COL9A1]
8: [GO_0007155, GO_0005201]; [COL24A1, COL4A3, COL4A6, COL5A3, COL9A1]
9: [GO_0007155, GO_0005201, GO_0005581]; [COL24A1, COL5A3]
10: [GO_0008544, GO_0005201, GO_0005581, GO_0001501, GO_0005584,
GO_0007605, GO_0005585]; [COL2A1]
11: [GO_0007605, GO_0005201]; [COL2A1, COL4A3]
12: [GO_0007605, GO_0005201, GO_0007155, GO_0008015, GO_0005587, GO_0005194,
GO_0008285, GO_0009405]; [COL4A3]
13: [GO_0007397, GO_0005201]; [COL3A1, COL5A3, COL9A1]
14: [GO_0007397, GO_0005201, GO_0005581]; [COL3A1, COL5A3]
15: [GO_0007397, GO_0005201, GO_0007155]; [COL5A3, COL9A1]
16: [GO_0007397, GO_0005201, GO_0005581, GO_0007155, GO_0007517, GO_0005588];
[COL5A3]
17: [GO_0008015, GO_0005201]; [COL3A1, COL4A3]
18: [GO_0008015, GO_0005201, GO_0005581, GO_0007397, GO_0005586]; [COL3A1]
Figure 7-16:

0: [GO_0000165, GO_0005194]; [MTMR1, MTMR2, MTMR3, MTMR4, MTMR6, MTMR7, MTMR8, FGFR1, FGFR2, FGFR3, FGFR4, RET, TEK, TIE1, COL1A2, COL2A1, COL24A1, COL27A1, COL2A1, COL3A1, COL4A1, COL4A2, COL4A3, COL4A6, COL5A3, COL9A1, COL9A2]
1: [GO_0004722, GO_0000165, GO_0005194, GO_0016787, GO_0006470, GO_0008372]; [MTMR1, MTMR2, MTMR6]
2: [GO_0004725, GO_0000165, GO_0005194, GO_0016787, GO_0006470]; [MTMR1, MTMR6, MTMR7]
3: [GO_0004725, GO_0000165, GO_0005194, GO_0004722, GO_0016787, GO_0006470, GO_0008372]; [MTMR1, MTMR6]
4: [GO_0016787, GO_0000165, GO_0005194, GO_0006470]; [MTMR1, MTMR2, MTMR3, MTMR6, MTMR7, MTMR8]
5: [GO_0006470, GO_0000165, GO_0005194]; [MTMR1, MTMR2, MTMR3, MTMR4, MTMR6, MTMR7, MTMR8]
6: [GO_0008372, GO_0000165, GO_0005194]; [MTMR1, MTMR2, MTMR6, RET]
7: [GO_0004727, GO_0000165, GO_0005194, GO_0016787, GO_0006470, GO_0008138]; [MTMR2, MTMR3]
8: [GO_0004727, GO_0000165, GO_0005194, GO_0004722, GO_0016787, GO_0006470, GO_0008372, GO_0007517, GO_0008151, GO_0008138]; [MTMR2]
9: [GO_0004727, GO_0000165, GO_0005194, GO_0004722, GO_0016787, GO_0006470, GO_0008372, GO_0004725, GO_0007517, GO_0008151, GO_0008138, GO_0005737, GO_0016020, GO_0004721, GO_0008270, GO_0006118, GO_0004713, GO_0004872, GO_0005007, GO_0005524, GO_0008201, GO_0016740, GO_0006468, GO_0016021, GO_0001501, GO_0008543, GO_0005887, GO_0016049, GO_0005624, GO_0004674, GO_0005007, GO_0005524, GO_0008201, GO_0016740, GO_0006468, GO_0016021, GO_0001501, GO_0008543, GO_0005887, GO_0016049, GO_0005624, GO_0004674, GO_0007259, GO_0005509, GO_0007156, GO_0007165, GO_0007497, GO_0005554, GO_0000004, GO_0004714, GO_0008603, GO_0005952, GO_0007166, GO_0007155, GO_0005198, GO_0007169, GO_0007267, GO_0007498, GO_0005201, GO_0005581, GO_0008147, GO_0005584, GO_0008544, GO_0007605, GO_0005585, GO_0007397, GO_0008015, GO_0005586, GO_0005587, GO_0030198, GO_0008285, GO_0009405, GO_0005588, GO_0005578, GO_0005594]; []
10: [GO_0007517, GO_0000165, GO_0005194]; [MTMR2, COL5A3]
11: [GO_0008151, GO_0000165, GO_0005194, GO_0008372]; [MTMR2, RET]
12: [GO_0005737, GO_0000165, GO_0005194, GO_0016787, GO_0006470, GO_0004727, GO_0008138, GO_0016020]; [MTMR3]
13: [GO_0016020, GO_0000165, GO_0005194]; [MTMR3, RET]
14: [GO_0016020, GO_0000165, GO_0005194, GO_0008372, GO_0008151, GO_0004713, GO_0004872, GO_0005524, GO_0016740, GO_0006468, GO_0016021, GO_0005509, GO_0007156, GO_0007165, GO_0007497, GO_0005554, GO_0000004, GO_0004714, GO_0008603, GO_0005952, GO_0007166, GO_0007155]; [RET]
15: [GO_0004721, GO_0000165, GO_0005194, GO_0006470]; [MTMR4, MTMR8]
16: [GO_0004721, GO_0000165, GO_0005194, GO_0016787, GO_0006470]; [MTMR8]
17: [GO_0008270, GO_0000165, GO_0005194, GO_0006470, GO_0004721]; [MTMR4]
18: [GO_0006118, GO_0000165, GO_0005194, GO_0004725, GO_0016787, GO_0006470]; [MTMR7]
19: [GO_0004713, GO_0000165, GO_0005194, GO_0004872, GO_0005524, GO_0016740, GO_0006468]; [FGFR1, FGFR2, FGFR3, FGFR4, RET]
20: [GO_0004872, GO_0000165, GO_0005194, GO_0006470, GO_0005524, GO_0016740, GO_0006468]; [FGFR1, FGFR2, FGFR3, FGFR4, RET, TEK, TIE1]
21: [GO_0005007, GO_0000165, GO_0005194, GO_0004713, GO_0004872, GO_0005524, GO_0016740, GO_0006468]; [FGFR1, FGFR2, FGFR3, FGFR4]
22: [GO_0008201, GO_0000165, GO_0005194, GO_0004713, GO_0004872, GO_0005524, GO_0005007, GO_0016740, GO_0006468, GO_0016021, GO_0001501, GO_0008543, GO_0005887, GO_0005624]; [FGFR1]
23: [GO_0016021, GO_0000165, GO_0005194, GO_0004713, GO_0004872, GO_0005524, GO_0016740, GO_0006468]; [FGFR1, FGFR2, RET]
24: [GO_0016021, GO_0000165, GO_0005194, GO_0004713, GO_0004872, GO_0005524, GO_0016740, GO_0006468, GO_0005007]; [FGFR1, FGFR2]
25: [GO_0000165, GO_0005194, GO_0004713, GO_0004872, GO_0005524, GO_0016740, GO_0006468, GO_0005007, GO_0001501, GO_0008543, GO_0005887, GO_0005624]; [FGFR1, FGFR3]
26: [GO_0001501, GO_0000165, GO_0005194]; [FGFR1, FGFR3, COL1A2, COL2A1, COL9A2]
27: [GO_0008543, GO_0000165, GO_0005194, GO_0004713, GO_0004872, GO_0005524, GO_0016740, GO_0006468, GO_0005007, GO_0005887]; [FGFR1, FGFR3, FGFR4]
28: [GO_0005887, GO_0000165, GO_0005194, GO_0004713, GO_0004872, GO_0005524, GO_0016740, GO_0006468]; [FGFR1, FGFR3, FGFR4, TEK, TIE1]
29: [GO_0004674, GO_0000165, GO_0005194, GO_0004713, GO_0004872, GO_0005524, GO_0016740, GO_0006468, GO_0005007, GO_0016021]; [FGFR2]
30: [GO_0007259, GO_0000165, GO_0005194, GO_0004713, GO_0004872, GO_0005524, GO_0016740, GO_0006468, GO_0005007, GO_0001501, GO_0008543, GO_0005887, GO_0016049]; [FGFR3]
31: [GO_0007165, GO_0000165, GO_0005194, GO_0004872, GO_0005524, GO_0016740,
Appendix: Concept Information of Figure 7-17-a, 7-17-b, 7-17-c, and 7-18

Figure 7-17-a:
0: [GO_0006470]; [AAC79117, AAD40368, CAA12271, AAC12865, AAC51682, AAC79118, AAH30779, BAA83025, AAC79119, AAF40203, AAF40204, AAF40205, BAA20826, AAH35609, AAC78841, AAH40012, AAL01037, CAD89918, AAC77820, AAH12399, BAA90964]
1: [GO_0004722, GO_0006470, GO_0016787, GO_0008372]; [AAC79117, AAD40368, CAA12271, AAC12865, AAC51682, AAC79118, AAH30779, BAA83025, AAC78841, AAH40012, AAL01037, CAD89918]
2: [GO_0004722, GO_0006470, GO_0016787]; [AAC79117, AAD40368, CAA12271, AAC78841, AAH40012, AAL01037, CAD89918]
3: [GO_0004722, GO_0006470, GO_0004725, GO_0016787, GO_0008372]; [AAC79117, AAD40368, CAA12271, AAC78841, AAH40012, AAL01037, CAD89918]
4: [GO_0016787, GO_0006470]; [AAC79117, AAD40368, CAA12271, AAC12865, AAC51682, AAC79118, AAH30779, BAA83025, AAC79119, AAF40203, AAF40204, AAF40205, BAA20826, AAC78841, AAH40012, AAL01037, CAD89918, AAC77820, AAH12399, BAA90964]
5: [GO_0004727, GO_0006470, GO_0016787, GO_0004725, GO_0008138]; [AAC12865, AAC51682, AAC79118, AAH30779, BAA83025, AAC79119, AAF40203, AAF40204, AAF40205, BAA20826, AAC78841, AAH40012, AAL01037, CAD89918, AAC77820, AAH12399, BAA90964]
6: [GO_0004727, GO_0006470, GO_0004725, GO_0016787, GO_0008138]; [AAC12865, AAC51682, AAC79118, AAH30779, BAA83025, AAC79119, AAF40203, AAF40204, AAF40205, BAA20826]
7: [GO_0004727, GO_0006470, GO_0004725, GO_0016787, GO_0008138, GO_0008151, GO_0005737, GO_0005717, GO_0008151, GO_0008138, GO_0005737, GO_0016020, GO_0004721, GO_0008270, GO_0006118]; []
8: [GO_0004727, GO_0006470, GO_0016787, GO_0004725, GO_0008138, GO_0016020]; [AAC79119, AAF40203, AAF40204, AAF40205, BAA20826]
9: [GO_0004721, GO_0006470]; [AAH35609, AAH12399, BAA90964]
10: [GO_0004721, GO_0006470, GO_0016787]; [AAH12399, BAA90964]
11: [GO_0008270, GO_0006470, GO_0004721]; [AAH35609]
12: [GO_0006118, GO_0006470, GO_0004725, GO_0016787]; [AAC77820]

Figure 7-17-b:
0: []; [AAA35835, AAA35836, AAA35837, AAA35838, AAA35839, AAA35840, AAA35958, AAA35959, AAA35960, AAA75007, CAA01958, CAA36101, CAA37015, CAA40400, CAA40401, CAA40402, CAA40403, CAA40404, CAA47375, CAA68679, CAA36147, AAA36152, AAA52449, AAA59470, AAA59471, AAA59472, AAA61188, AAA68514, AAC41933, AAC41934, AAC41935, AAC41936, AAC41937, AAC41938, AAD14392, AAD31560, AAD31561, AAD31562, AAD31565, AAD31567, AAF43273, AAF43274, AAK94205, AAK94206, AAK94207, AAK94208, AAK94209, AAM74056, BAA89296, BAA89297, BAA89298, BAA89299, BAA89300, BAA89301, BAC45037, CAA39654, CAA76643, CAA96492, AAA52450, AAA58470, AAA63209, AAA67781, AAB33323, AAF63380, AAK54727, AAK54729, CAA59344, AAA36524, AAA36786, AAA60266, AAH03072, AAH04257, CAA31408, CAA33333, CAA33787, CAA39792, AAA61139, BAC45250, CAA43290]
1: [GO_0004713, GO_0005524, GO_0006468]; [AAA35835, AAA35836, AAA35837, AAA35840, AAA35958, AAA35959, AAA35960, AAA75007, CAA01958, CAA36101, CAA37015, CAA40400, CAA40401, CAA40402, CAA40403, CAA40404, CAA47375, CAA68679, AAA36147, AAA36152, AAA52449, AAA59470, AAA59471, AAA59472, AAA61188, AAA68514, AAC41933, AAC41934, AAC41935, AAC41936, AAC41937, AAC41938, AAD14392, AAD31560, AAD31561, AAD31562, AAD31565, AAD31567, AAF43273, AAF43274, AAK94205, AAK94206, AAK94207, AAK94208, AAK94209, AAM74056, BAA89296, BAA89297, BAA89298, BAA89299, BAA89300, BAA89301, BAC45037, CAA39654, CAA76643, CAA96492, AAA52450, AAA58470, AAA63209, AAA67781, AAB33323, AAF63380, AAK54727, AAK54729, CAA59344, AAA36524, AAA36786, AAA60266, AAH03072, AAH04257, CAA31408, CAA33333, CAA33787, CAA39792, AAA61139, BAC45250, CAA43290]
2: [GO_0004872]; [AAA35835, AAA35836, AAA35837, AAA35838, AAA35839, AAA35840, AAA35958, AAA35959, AAA35960, AAA75007, CAA01958, CAA36101, CAA37015, CAA40400, CAA40401, CAA40402, CAA40403, CAA40404, CAA47375, CAA68679, AAA36147, AAA36152, AAA52449, AAA59470, AAA59471, AAA59472, AAA61188, AAA68514, AAC41933, AAC41934, AAC41935, AAC41936, AAC41937, AAC41938, AAD14392, AAD31560, AAD31561, AAD31562, AAD31565, AAD31567, AAF43273, AAF43274, AAK94205, AAK94206, AAK94207, AAK94208, AAK94209, AAM74056, BAA89296, BAA89297, BAA89298, BAA89299, BAA89300, BAA89301, BAC45037, CAA39654, CAA76643, CAA96492, AAA52450, AAA58470, AAA63209, AAA67781, AAB33323, AAF63380, AAK54727, AAK54729, CAA59344, AAA36524, AAA36786, AAA60266, AAH03072, AAH04257, CAA31408, CAA33333, CAA33787, CAA39792]
AAB33323, AAF63380, AAK54727, CAA59334, AAA63208, AAB59389, AAH11847, AAK34949, CAA40490, CAA74200, CAB89250, AAA36524, AAA36786, AAA60266, AAH03072, AAH04257, CAA31408, CAA33333, CAA33787, CAA39792, AAA61139, BAC45250, CAA43290

3: [GO_0004872, GO_0004713, GO_0005524, GO_0016740, GO_0006468]; [AAA35835, AAA35836, AAA35837, AAA35840, AAA35958, AAA35959, AAA35960, AAA75007, CAA01958, CAA36101, CAA37015, CAA40400, CAA40401, CAA40402, CAA40403, CAA47375, CAA68679, AAA36147, AAA36152, AAA52449, AAA59470, AAA59471, AAA59472, AAA61188, AAA68514, AAC41933, AAC41934, AAC41935, AAC41936, AAC41937, AAC41938, AAD14392, AAD31560, AAD31561, AAD31562, AAD31565, AAF43273, AAF43274, AAK94205, AAK94206, AAK94207, AAK94208, AAM74056, BAA89296, BAA89297, BAA89298, BAA89299, BAA89300, BAA89301, CAA37014, CAA39654, CAA76643, CAA96492, AAA52450, AAAS8470, AAA63209, AAA67781, AAB33323, AAF63380, AAK54727, CAA59334, AAA63208, AAB59389, AAH11847, AAK34949, CAA40490, CAA74200, CAB89250, AAA36524, AAA36786, AAA60266, AAH03072, AAH04257, CAA31408, CAA3333, CAA33787, CAA39792]

4: [GO_0005007, GO_0004872]; [AAA35835, AAA35836, AAA35837, AAA35838, AAA35839, AAA35840, AAA35958, AAA35959, AAA35960, AAA75007, CAA01958, CAA36101, CAA37015, CAA40400, CAA40401, CAA40402, CAA40403, CAA40404, CAA47375, CAA68679, AAA36147, AAA36152, AAA52449, AAA59470, AAA59471, AAA59472, AAA61188, AAA68514, AAC41933, AAC41934, AAC41935, AAC41936, AAC41937, AAC41938, AAD14392, AAD31560, AAD31561, AAD31562, AAD31565, AAF43273, AAF43274, AAK94205, AAK94206, AAK94207, AAK94208, AAK94209, AAM74056, BAA89296, BAA89297, BAA89298, BAA89299, BAA89300, BAA89301, CAA37014, CAA39654, CAA76643, CAA96492, AAA52450, AAAS8470, AAA63209, AAA67781, AAB33323, AAF63380, AAK54727, CAA59334, AAA63208, AAB59389, AAH11847, AAK34949, CAA40490, CAA74200, CAB89250, AAA36524, AAA36786, AAA60266, AAH03072, AAH04257, CAA31408, CAA3333, CAA33787, CAA39792]

5: [GO_0005007, GO_0004713, GO_0005524, GO_0016740, GO_0006468]; [AAA35835, AAA35836, AAA35837, AAA35838, AAA35839, AAA35840, AAA35958, AAA35959, AAA35960, AAA75007, CAA01958, CAA36101, CAA37015, CAA40400, CAA40401, CAA40402, CAA40403, CAA47375, CAA68679, AAA36147, AAA36152, AAA52449, AAA59470, AAA59471, AAA59472, AAA61188, AAA68514, AAC41933, AAC41934, AAC41935, AAC41936, AAC41937, AAC41938, AAD14392, AAD31560, AAD31561, AAD31562, AAD31565, AAF43273, AAF43274, AAK94205, AAK94206, AAK94207, AAK94208, AAK94209, AAM74056, BAA89296, BAA89297, BAA89298, BAA89299, BAA89300, BAA89301, CAA37014, CAA39654, CAA76643, CAA96492, AAA52450, AAAS8470, AAA63209, AAA67781, AAB33323, AAF63380, AAK54727, AACKA39792, AACKA59334, AAA63208, AAB59389, AAH11847, CAA40490, CAA74200]

6: [GO_0005524, GO_0006468]; [AAA35835, AAA35836, AAA35837, AAA35840, AAA35958, AAA35959, AAA35960, AAA75007, CAA01958, CAA36101, CAA37015, CAA40400, CAA40401, CAA40402, CAA40403, CAA47375, CAA68679, AAA36147, AAA36152, AAA52449, AAA59470, AAA59471, AAA59472, AAA61188, AAA68514, AAC41933, AAC41934, AAC41935, AAC41936, AAC41937, AAC41938, AAD14392, AAD31560, AAD31561, AAD31562, AAD31565, AAF43273, AAF43274, AAK94205, AAK94206, AAK94207, AAK94208, AAK94209, AAM74056, BAA89296, BAA89297, BAA89298, BAA89299, BAA89300, BAA89301, CAA37014, CAA39654, CAA76643, CAA96492, AAA52450, AAAS8470, AAA63209, AAA67781, AAB33323, AAF63380, AAK54727, AACKA39792, AACKA59334, AAA63208, AAB59389, AAH11847, CAA40490, CAA74200]
AAA75007, CAA01958, CAA36101, CAA37015, CAA40400, CAA40401, CAA40402, CAA40403, CAA40404, CAA47375, CAA68679, AAA6147, AAA36152, AAA52449, AAA59470, AAA59471, AAA59472, AAA61188, AAA68514, AAC41933, AAC41934, AAC41935, AAC41936, AAC41937, AAC41938, AAD14392, AAD31560, AAD31561, AAD31562, AAD31565, AAD31567, AAF43273, AAF43274, AAK94205, AAK94206, AAK94207, AAK94208, AAK94209, AAM74056, BAA89296, BAA89297, BAA89298, BAA89299, BAA89300, BAA89301, CAA37014, CAA39654, CAA76643, CAA96492, AAA36524, AAA36786, AAA60266, AAH03072, AAH04257, CAA31408, CAA33333, CAA33787, CAA39792

11: [GO_0016021, GO_0004872, GO_0005007]; [AAA35835, AAA35836, AAA35837, AAA35838, AAA35839, AAA35840, AAA35958, AAA35959, AAA35960, AAA75007, CAA01958, CAA36101, CAA37015, CAA40400, CAA40401, CAA40402, CAA40404, CAA47375, CAA68679, AAA61188, AAA68514, AAC41933, AAC41934, AAC41935, AAC41936, AAC41937, AAC41938, AAD14392, AAD31560, AAD31561, AAD31562, AAD31565, AAD31567, AAF43273, AAF43274, AAK94205, AAK94206, AAK94207, AAK94208, AAK94209, AAM74056, BAA89296, BAA89297, BAA89298, BAA89299, BAA89300, BAA89301, CAA37014, CAA39654, CAA76643, CAA96492]

12: [GO_0016021, GO_0004713, GO_0005524, GO_0006468, GO_0004872, GO_0016740, GO_0005007]; [AAA35835, AAA35836, AAA35837, AAA35840, AAA35958, AAA35959, AAA35960, AAA75007, CAA01958, CAA36101, CAA37015, CAA40400, CAA40401, CAA40404, CAA47375, CAA68679, AAA61188, AAA68514, AAC41933, AAC41934, AAC41935, AAC41936, AAC41937, AAC41938, AAD14392, AAD31560, AAD31561, AAD31562, AAD31565, AAD31567, AAF43273, AAF43274, AAK94205, AAK94206, AAK94207, AAK94208, AAK94209, AAM74056, BAA89296, BAA89297, BAA89298, BAA89299, BAA89300, BAA89301, CAA37014, CAA39654, CAA76643, CAA96492]

13: [GO_0000165, GO_0004713, GO_0005524, GO_0006468, GO_0004872, GO_0016740, GO_0005007, GO_0001501, GO_0008543, GO_0005887]; [AAA35835, AAA35836, AAA35837, AAA35840, AAA35958, AAA35959, AAA35960, AAA75007, CAA01958, CAA36101, CAA37015, CAA40400, CAA40401, CAA40404, CAA47375, CAA68679, AAA359471, AAA59472, AAA61188, AAA68514, AAC41933, AAC41934, AAC41935, AAC41936, AAC41937, AAC41938, AAD14392, AAD31560, AAD31561, AAD31562, AAD31565, AAD31567, AAF43273, AAF43274, AAK94205, AAK94206, AAK94207, AAK94208, AAK94209, AAM74056, BAA89296, BAA89297, BAA89298, BAA89299, BAA89300, BAA89301, CAA37014, CAA39654, CAA76643, CAA96492]

14: [GO_0008543, GO_0004713, GO_0005524, GO_0006468, GO_0004872, GO_0016740, GO_0005007, GO_0005887]; [AAA35835, AAA35836, AAA35837, AAA35840, AAA35958, AAA35959, AAA35960, AAA75007, CAA01958, CAA36101, CAA37015, CAA40400, CAA40401, CAA40404, CAA47375, CAA68679, AAA52450, AAA58470, AAA63209, AAA67781, AAB33323, AAF63380, AAK54727, AAK54729, CAA59334]

122
15: [GO_0005887, GO_0004872]; [AAA35835, AAA35836, AAA35837, AAA35840, AAA35958, AAA35959, AAA35960, AAA75007, CAA01958, CAA36101, CAA37015, CAA40400, CAA40401, CAA40402, CAA40403, CAA40404, CAA47375, CAA68679, AAA52450, AAA58470, AAA63209, AAA67781, AAB33323, AAF63380, AAK54727, AAK54729, CAA59334, AAA63208, AAB59389, AH11847, CAA40490, CAA74200, AAA61139, BAC45250, CAA43290]

16: [GO_0005887, GO_0004872, GO_0005524, GO_0006468, GO_0016740]; [AAA35835, AAA35836, AAA35837, AAA35840, AAA35958, AAA35959, AAA35960, AAA75007, CAA01958, CAA36101, CAA37015, CAA40400, CAA40401, CAA40402, CAA40403, CAA40404, CAA47375, CAA68679, AAA52450, AAA58470, AAA63209, AAA67781, AAB33323, AAF63380, AAK54727, AAK54729, CAA59334, AAA63208, AAB59389, AH11847, CAA40490, CAA74200, AAA61139, CAA43290]

17: [GO_0016049, GO_0004872, GO_0005007]; [AAA35838, AAA35839, AAA52450, AAA63208, AAA67781, AAB33323, AAF63380, AAK54727, AAK54729, CAA59334]

18: [GO_0016049, GO_0004713, GO_0005524, GO_0006468, GO_0016740, GO_0005007, GO_0001501, GO_0008201, GO_0016021, GO_0008543, GO_0005887, GO_0007259]; [AAA52450, AAA58470, AAA63209, AAA67781, AAB33323, AAF63380, AAK54727, AAK54729, CAA59334]

19: [GO_0016049, GO_0004713, GO_0005524, GO_0006468, GO_0004872, GO_0016740, GO_0005007, GO_0001501, GO_0008543, GO_0005887, GO_0005624, GO_0004674, GO_0004714, GO_0008603, GO_0005952, GO_0007156, GO_0007165, GO_0005194, GO_0007155, GO_0016020, GO_0005198, GO_0007169, GO_0007498]; []

20: [GO_0016049, GO_0004872, GO_0005007, GO_0016021, GO_0005624]; [AAA35838, AAA35839]

21: [GO_0004674, GO_0004713, GO_0005524, GO_0006468]; [BAC45037]

22: [GO_0005509, GO_0004713, GO_0005524, GO_0006468, GO_0004872, GO_0016740, GO_0016021, GO_0001501, GO_0008543, GO_0005887, GO_0005624, GO_0004674, GO_0004714, GO_0008603, GO_0005952, GO_0007166, GO_0005194, GO_0007155, GO_0016020, GO_0005198, GO_0007169, GO_0007498]; [AAA36524, AAA36786, AAA60266, AH03072, AHO4257, CAA31408, CAA33333, CAA33787, CAA39792]

23: [GO_0007165, GO_0004872, GO_0004714]; [AAA36524, AAA36786, AAA60266, AHO3072, AHO4257, CAA31408, CAA33333, CAA33787, CAA39792, AAA61139, BAC45250, CAA43290]

24: [GO_0007165, GO_0004872, GO_0005524, GO_0006468, GO_0016740, GO_0004714]; [AAA36524, AAA36786, AAA60266, AHO3072, AHO4257, CAA31408, CAA33333, CAA33787, CAA39792, AAA61139, CAA43290]

25: [GO_0007165, GO_0004872, GO_0005887, GO_0004714]; [AAA61139, BAC45250, CAA43290]
Figure 7-17-c:

0: [ ]; [AAA51844, AAA51846, AAA59994, AAA60041, AAB59374, AAB69977, AAB93981, CAA23761, CAA39142, CAA68709, CAA98969, AAL02227, AAL86699, AAN03650, BAB13947, BAB47499, AAA51992, AAA51993, AAA51995, AAA51997, AAA52039, AAA52051, AAA52052, AAA58428, AAA60150, AAB59576, AAC41772, AAH07252, AAL13166, AAP35869, CAA25092, CAA29604, CAA29605, CAA30731, CAA32030, CAA34488, CAA98968, AAA52003, AAA52040, AAH28178, CAA25879, CAA29886, CAA30229, CAA32583, CAA33387, AAA52006, AAA52042, AAA53098, CAA29075, CAA68698, AAA53097, AAA53099, AAA58422, CAA29076, CAA29089, CAA31275, AAA18942, AAA18943, AAA21610, AAA51556, AAA52044, AAB19637, BAA25064, CAA56335, CAC36101, AAA16338, AAA19569, AAB19038, AAB19039, BAA04809, CAA20120, CAA22265, CAB89263, CAB96748, AAF59902, AAA53474, AAA53475, AAC33527, AAC33528, CAA38276, CAA38277, CAC19501, CAC19502, AAA80977, AAC33512, CAB81611]

1: [GO_0005201]; [AAA51844, AAA51846, AAA59994, AAA60041, AAB59374, AAB69977, AAB93981, CAA23761, CAA39142, CAA68709, CAA98969, AAN03650, BAB13947, BAB47499, AAA51992, AAA51993, AAA51995, AAA51997, AAA52039, AAA52051, AAA52052, AAA58428, AAA60150, AAB59576, AAC41772, AAH07252, AAL13166, AAP35869, CAA25092, CAA29604, CAA29605, CAA30731, CAA32030, CAA34488, CAA98968, AAA52003, AAA52040, AAH28178, CAA25879, CAA29886, CAA30229, CAA32583, CAA33387, AAA52006, AAA52042, AAA53098, CAA29075, CAA68698, AAA53097, AAA53099, AAA58422, CAA29076, CAA29089, CAA31275, AAA18942, AAA18943, AAA21610, AAA51556, AAA52044, AAB19637, BAA25064, CAA56335, CAC36101, AAA16338, AAA19569, AAB19038, AAB19039, BAA04809, CAA20120, CAA22265, CAB89263, CAB96748, AAF59902, AAA53474, AAA53475, AAC33527, AAC33528, CAA38276, CAA38277, CAC19501, CAC19502, AAA80977, AAC33512, CAB81611]

2: [GO_0005581, GO_0005201]; [AAA51844, AAA51846, AAA59994, AAA60041, AAB59374, AAB69977, AAB93981, CAA23761, CAA39142, CAA68709, CAA98969, AAN03650, BAB13947, BAB47499, AAA51992, AAA51993, AAA51995, AAA51997, AAA52039, AAA52051, AAA52052, AAA58428, AAA60150, AAB59576, AAC41772, AAH07252, AAL13166, AAP35869, CAA25092, CAA29604, CAA29605, CAA30731, CAA32030, CAA34488, CAA98968, AAA52003, AAA52040, AAH28178, CAA25879, CAA29886, CAA30229, CAA32583, CAA33387, AAA52006, AAA52042, AAA53098, CAA29075, CAA68698, AAA53097, AAA53099, AAA58422, CAA29076, CAA29089, CAA31275, AAA18942, AAA18943, AAA21610, AAA51556, AAA52044, AAB19637, BAA25064, CAA56335, CAC36101, AAA16338, AAA19569, AAB19038, AAB19039, BAA04809, CAA20120, CAA22265, CAB89263, CAB96748, AAF59902, AAA53474, AAA53475, AAC33527, AAC33528, CAA38276, CAA38277, CAC19501, CAC19502, AAA80977, AAC33512, CAB81611]
CAA32030, CAA34488, CAA98968, AAA52003, AAA52040, AH28178, CAA25879, CAA29886, CAA30229, CAA32583, CAA33387, AAF59902
3: [GO_0008147, GO_0005201, GO_0005581, GO_0001501, GO_0005584]; [AAA51844, AAA51846, AAA59994, AAA60041, AAB59374, AAB59377, AAB93981, CAA23761, CAA39142, CAA68709, CAA98969, AAA51992, AAA51993, AAA51995, AAA51997, AAA52039, AAA52051, AAA52052, AAA58428, AAA60150, AAB59576, AAC41772, AAL13166, CAA25092, CAA29604, CAA29605, CAA30731, CAA32030, CAA34488, CAA98968]
4: [GO_0001501, GO_0005201]; [AAA51844, AAA51846, AAA59994, AAA60041, AAB59374, AAB69977, CAA23761, CAA39142, CAA68709, CAA98969, AAA51992, AAA51993, AAA51995, AAA51997, AAA52039, AAA52051, AAA52052, AAA58428, AAA60150, AAB59576, AAC41772, AAL13166, CAA25092, CAA29604, CAA29605, CAA30731, CAA32030, CAA34488, CAA98968, AAA80977, AAC33512, CAA81611]
5: [GO_0005198, GO_0007155]; [AAL02227, AAL86699]
6: [GO_0005198, GO_0005201, GO_0005581, GO_0008147, GO_0001501, GO_0005584, GO_0007155, GO_0008544, GO_0007605, GO_0005585, GO_0007397, GO_0008015, GO_0005586, GO_0005587, GO_0007517, GO_0005587, GO_0005194, GO_0008285, GO_0009405, GO_0000004, GO_0005578, GO_0005594]; []
7: [GO_0007155]; [AAL02227, AAL86699, AAN03650, AAA18942, AAA18943, AAA21610, AAA51556, AAA52044, AAB19637, BAA25064, CAA56335, CAC36101, AAA16338, AAA19569, AAB19039, BAA04809, CAA20120, CAA22265, CAA98969, CAA98968, AAA80977, AAC33512, CAA81611]
8: [GO_0007155, GO_0005201]; [AAN03650, AAA18942, AAA18943, AAA21610, AAA51556, AAA52044, AAB19637, BAA25064, CAA56335, CAC36101, AAA16338, AAA19569, AAB19039, BAA04809, CAA20120, CAA22265, CAA98969, CAA98968, AAA80977, AAC33512, CAA81611]
9: [GO_0007155, GO_0005201, GO_0005581]; [AAN03650, AAF59902]
10: [GO_0008544, GO_0005201, GO_0005581, GO_0005584, GO_0007605, GO_0005585]; [AAA51992, AAA51993, AAA51995, AAA51997, AAA52039, AAA52051, AAA52052, AAA58428, AAA60150, AAB59576, AAC41772, AAL13166, CAA25092, CAA29604, CAA29605, CAA30731, CAA32030, CAA34488, CAA98968, AAA18942, AAA18943, AAA21610, AAA51556, AAA52044, AAB19637, BAA25064, CAA56335, CAC36101]
12: [GO_0007605, GO_0005201, GO_0007155, GO_0008015, GO_0005587, GO_0005194, GO_0008285, GO_0009405]; [AAA18942, AAA18943, AAA21610, AAA51556, AAA52044, AAB19637, BAA25064, CAA56335, CAC36101]
13: [GO_0007397, GO_0005201]; [AAA52003, AAA52040, CAA25879, CAA29886, CAA30229, CAA32583, CAA33387, AAF59902, AAA53474, AAA53475, AAC33527, AAC33528, CAA38276, CAA38277, CAC19501, CAC19502]
14: [GO_0007397, GO_0005201, GO_0005581]; [AAA52003, AAA52040, CAA25879, CAA29886, CAA30229, CAA33387, AAF59902, AAA52044, AAB19637, BAA25064, CAA56335, CAC36101]
15: [GO_0007397, GO_0005201, GO_0007155]; [AAA52003, AAA53474, AAA53475, AAC33527, AAC33528, CAA38276, CAA38277, CAC19501, CAC19502]
16: [GO_0007397, GO_0005201, GO_0005581, GO_0007155, GO_0007517, GO_0005588]; [AAF59902]
17: [GO_0008015, GO_0005201]; [AAA52003, AAA52040, CAA25879, CAA29886, CAA30229, CAA32583, CAA33387, AAA18942, AAA18943, AAA21610, AAA51556, AAA52044, AAB19637, BAA25064, CAA56335, CAC36101]
18: [GO_0008015, GO_0005201, GO_0005581, GO_0007397, GO_0005586]; [AAA52003, AAA52040, CAA25879, CAA29886, CAA30229, CAA32583, CAA33387]
19: [GO_0005587, GO_0005201]; [AAA52006, AAA52042, AAA53098, CAA29075, CAA68698, AAA53097, AAA53099, AAA58422, CAA29076, CAA29098, CAA31275, AAA18942, AAA18943, AAA21610, AAA51556, AAA52044, AAB19637, BAA25064, CAA56335, CAC36101, AAA16338, AAA19569, AAB19038, AAB19039, BAA04809, CAA20120, CAA22265, CAB89263, CAB96748]
20: [GO_0005587, GO_0005201, GO_0007155, GO_0005194]; [AAA18942, AAA18943, AAA21610, AAA51556, AAA52044, AAB19637, BAA25064, CAA56335, CAC36101, AAA16338, AAA19569, AAB19038, AAB19039, BAA04809, CAA20120, CAA22265, CAB89263, CAB96748]
21: [GO_0005587, GO_0005201, GO_0005587]; [AAA53097, AAA53099, AAA58422, CAA29076, CAA29098, CAA31275, AAA18942, AAA18943, AAA21610, AAA51556, AAA52044, AAB19637, BAA25064, CAA56335, CAC36101, AAA16338, AAA19569, AAB19038, AAB19039, BAA04809, CAA20120, CAA22265, CAB89263, CAB96748]
22: [GO_0005587, GO_0005201, GO_0007155, GO_0005587, GO_0005194]; [AAA16338, AAA19569, AAB19038, AAB19039, BAA04809, CAA20120, CAA22265, CAB89263, CAB96748]
23: [GO_0005587, GO_0005201, GO_0007155, GO_0007397, GO_0005587, GO_0005594]; [AAA53474, AAA53475, AAC33527, AAC33528, CAA38276, CAA38277, CAC19501, CAC19502]
24: [GO_0005578, GO_0005201, GO_0005594]; [AAA53474, AAA53475, AAC33527, AAC33528, CAA38276, CAA38277, CAC19501, CAC19502, AAA80977, AAC33512, CAB81611]
25: [GO_0005578, GO_0005201, GO_0001501, GO_0005594]; [AAA80977, AAC33512, CAB81611]
Figure 7-18:
0: [AAA51844, AAA51846, AAA59994, AAA60041, AAB59374, AAB69977, AAB93981, CAA23761, CAA39142, CAA68709, CAA98969, AALO2227, AAL86699, AAN03650, BAB13947, BABB47999, AAA51992, AAA51993, AAA51995, AAA51997, AAA52039, AAA52051, AAA52052, AAA58428, AAA60150, AAB59576, AAC14772, AAH07252, AAL13166, AAP35869, CAA25092, CAA29604, CAA29605, CAA30731, CAA32030, CAA34488, CAA98968, AAA52003, AAA52040, AAH28178, CAA25879, CAA29886, CAA30229, CAA32583, CAA33387, AAA52006, AAA52042, AAA53098, CAA29075, CAA68698, AAA53097, AAA53099, AAA58422, CAA29076, CAA29098, CAA31275, AAA18942, AAA18943, AAA21610, AAA51556, AAA52044, AAB19637, BAA25064, CAA56335, CAC36101, AAA16338, AAA19569, AAB19038, AAB19039, BAA04809, CAA20120, CAA22265, CAA98263, CAA96748, AAF59902, AAA53474, AAA53475, AAC33527, AAC33528, CAA38276, CAA38277, CAC19501, CAC19502, AAA80977, AAC33512, CBB16111, CAC79117, AAD0368, CAA12271, AAD12865, AAD15682, AAD21198, AAD30779, BAA83025, AAD39119, AAD40302, AAD40204, AAD40205, BAA20826, AAD35609, AAD78841, AAH40012, AAD01037, CA89918, AAC77820, AADH12399, BAA90964, AAA38535, AAA38536, AAA38537, AAA38538, AAA38539, AAA38540, AAA38549, AAA38590, AAA38590, AAA75007, CAA1958, CAA36101, CAA37015, CAA40040, AAD4001, CAA4004, CAA44735, CAA68679, AAA36147, AAA36152, AAA52449, AAA59470, AAA59471, AAA59472, AAA61118, AAA68514, AAC41934, AAC41935, AAC41936, AAC41937, AAC41938, AAD14392, AAD31560, AAD31561, AAD31562, AAD31565, AAD31567, AAF43273, AAF43274, AAK92920, AAK99206, AAK99207, AAK99208, AAK99209, AAM74056, BAA89296, BAA89297, BAA89298, BAA89299, BAA89300, BAA89301, BAC45037, CAA3764, CAA39654, CAA76643, CAA96492, AAA52450, AAA58470, AAA63209, AAA66781, AAD3323, AAF63380, AAK54727, AAK54729, CAA5934, AAA63208, BAA59389, AAH11487, AAD43499, CAA40409, CAA47200, BAA89250, AAA36524, AAA36786, AAA60266, AAH03072, AAH04257, CAA31408, CAA3333, CAA33787, CAA39792, AAA61139, BAC45250, CAA43290; 1: [GO_0005201]; [AAA51844, AAA51846, AAA59994, AAA60041, AAB59374, AAB69977, AAB93981, CAA23761, CAA39142, CAA68709, CAA98969, AAL02227, AAL86699, AAN03650, BAB13947, BABB47999, AAA51992, AAA51993, AAA51995, AAA51997, AAA52039, AAA52051, AAA52052, AAA58428, AAA60150, AAB59576, AAC14772, AAH07252, AAL13166, AAP35869, CAA25092, CAA29604, CAA29605, CAA30731, CAA32030, CAA34488, CAA98968, AAA52003, AAA52040, AAH28178, CAA25879, CAA29886, CAA30229, CAA32583, CAA33387, AAA52006, AAA52042, AAA53098, CAA29075, CAA68698, AAA53097, AAA53099, AAA58422, CAA29076, CAA29098, CAA31275, AAA18942, AAA18943, AAA21610, AAA51556, AAA52044, AAB19637, BAA25064, CAA56335, CAC36101, AAA16338, AAA19569, AAB19038, AAB19039, BAA04809, CAA20120, CAA22265, CAA98263, CAA96748, AAF59902, AAA53474, AAA53475, AAC33527]
AAC33528, CAA38276, CAA38277, CAC19501, CAC19502, AAA80977, AAC33512, CAB81611]
2: [GO_0005581, GO_0005201]; [AAA51844, AAA51846, AAA59994, AAA60041, AAB59374, AAB69977, AAB93981, CAA23761, CAA39142, CAA68709, CAA98969, AAN03650, BAB13947, BAB47499, AAA51992, AAA51993, AAA51995, AAA51997, AAS52039, AAS52051, AAS52052, AAS58428, AAA60150, AAB59576, AAC41772, AAH07252, AAL13166, AAP35869, CAA25092, CAA29604, CAA29605, CAA30731, CAA32030, CAA34488, CAA98968, AAA52003, AAA52040, AAH28178, CAA25879, CAA29886, CAA30229, CAA32583, CAA33387, AAF59902]
3: [GO_0008147, GO_0005201, GO_0005581, GO_0001501, GO_0005584]; [AAA51844, AAA51846, AAA59994, AAA60041, AAB59374, AAB69977, AAB93981, CAA23761, CAA39142, CAA68709, CAA98969, AAA51992, AAA51993, AAA51995, AAA51997, AAS52039, AAS52051, AAS52052, AAS58428, AAA60150, AAB59576, AAC41772, AAL13166, CAA25092, CAA29604, CAA29605, CAA30731, CAA32030, CAA34488, CAA98968]
4: [GO_0001501]; [AAA51844, AAA51846, AAA59994, AAA60041, AAB59374, AAB69977, AAB93981, CAA23761, CAA39142, CAA68709, CAA98969, AAA51992, AAA51993, AAA51995, AAS52039, AAS52051, AAS52052, AAS58428, AAA60150, AAB59576, AAC41772, AAL13166, CAA25092, CAA29604, CAA29605, CAA30731, CAA32030, CAA34488, CAA98968, AAA80977, AAC33512, CAB81611, AAA35835, AAA35836, AAA35837, AAA35840, AAA35858, AAA35958, AAA35959, AAA35960, AAA75007, CAA01958, CAA36101, CAA37015, CAA40400, CAA40401, CAA40402, CAA40403, CAA40404, CAA47375, CAA68679, AAA52450, AAA58470, AAA63209, AAA67781, AAB33323, AAF63380, AAK54727, AAK54729, CAA59334]
5: [GO_0001501, GO_0005201]; [AAA51844, AAA51846, AAA59994, AAA60041, AAB59374, AAB69977, AAB93981, CAA23761, CAA39142, CAA68709, CAA98969, AAA51992, AAA51993, AAA51995, AAA51997, AAS52039, AAS52051, AAS52052, AAS58428, AAA60150, AAB59576, AAC41772, AAL13166, CAA25092, CAA29604, CAA29605, CAA30731, CAA32030, CAA34488, CAA98968, AAA80977, AAC33512, CAB81611, AAA35835, AAA35836, AAA35837, AAA35840, AAA35858, AAA35958, AAA35959, AAA35960, AAA75007, CAA01958, CAA36101, CAA37015, CAA40400, CAA40401, CAA40402, CAA40403, CAA40404, CAA47375, CAA68679, AAA52450, AAA58470, AAA63209, AAA67781, AAB33323, AAF63380, AAK54727, AAK54729, CAA59334]
6: [GO_0005198]; [AAL02227, AAL86699, AAA61139, CAA43290]
7: [GO_0005198, GO_0005201, GO_0005581, GO_0008147, GO_001501, GO_0005584, GO_007155, GO_008544, GO_007605, GO_005585, GO_007397, GO_008015, GO_005586, GO_005587, GO_0030198, GO_005194, GO_008285, GO_009405, GO_007517, GO_005588, GO_000004, GO_005578, GO_005594, GO_004722, GO_004725, GO_0016787, GO_006470, GO_008372, GO_004727, GO_008151, GO_008138, GO_005737, GO_0016020, GO_004721, GO_008270, GO_006118, GO_004713, GO_004872, GO_005007, GO_005524, GO_008201, GO_0016740, GO_006468, GO_0016021, GO_0000165, GO_008543, GO_005887, GO_0016049, GO_005624, GO_004674, GO_007259, GO_005509, GO_007156, GO_007165,
GO_0007497, GO_0005554, GO_0004714, GO_0008603, GO_0005952, GO_0007166, GO_0007169, GO_0007267, GO_0007498;
8: [GO_0007155]; [AAL02227, AAL86699, AAN03650, AAA18942, AAA18943, AAA21610, AAA51556, AAA52044, AAB19637, BAA25064, CAA56335, CAC36101, AAA16338, AAA19569, AAB19038, AAB19039, BAA04809, CAA20120, CAA22265, CABA9263, CAB96748, AAF59902, AAA53474, AAA53475, AAC33527, AAC33528, CAA38276, CAA38277, CAC19501, CAC19502, AAA36524, AAA36786, AAA60266, AAH03072, AAH04257, CAA31408, CAA33333, CAA33787, CAA39792]
9: [GO_0007155, GO_0005201]; [AAN03650, AAA18942, AAA18943, AAA21610, AAA51556, AAA52044, AAB19637, BAA25064, CAA56335, CAC36101, AAA16338, AAA19569, AAB19038, AAB19039, BAA04809, CAA20120, CAA22265, CABA9263, CAB96748, AAF59902, AAA53474, AAA53475, AAC33527, AAC33528, CAA38276, CAA38277, CAC19501, CAC19502]
10: [GO_0007155, GO_0005201, GO_0005581]; [AAN03650, AAF59902]
11: [GO_0007155, GO_0005198]; [AAL02227, AAL86699]
12: [GO_0008544, GO_0005201, GO_0005581, GO_0008147, GO_0001501, GO_0005584, GO_0007605, GO_0005585]; [AAA51992, AAA51993, AAA51995, AAA51997, AAA52039, AAA52051, AAA58428, AAA60150, AAB59576, AAC41772, AAL13166, CAA25092, CAA29604, CAA30731, CAA32030, CAA34488, CAA98968]
13: [GO_0007605, GO_0005201]; [AAA51992, AAA51993, AAA51995, AAA51997, AAA52039, AAA52051, AAA58428, AAA60150, AAB59576, AAC41772, AAL13166, CAA25092, CAA29604, CAA30731, CAA32030, CAA34488, CAA98968, AAA18942, AAA18943, AAA21610, AAA51556, AAA52044, AAB19637, BAA25064, CAA56335, CAC36101]
14: [GO_0007605, GO_0005201, GO_0005581, GO_0008015, GO_0005587, GO_0005194, GO_0008285, GO_0009405]; [AAA18942, AAA18943, AAA21610, AAA51556, AAA52044, AAB19637, BAA25064, CAA56335, CAC36101]
15: [GO_0007397, GO_0005201]; [AAA52003, AAA52040, CAA25879, CAA29886, CAA30229, CAA32583, CAA33387, AAF59902, AAA53474, AAA53475, AAC33527, AAC33528, CAA38276, CAC19501, CAC19502]
16: [GO_0007397, GO_0005201, GO_0005581]; [AAA52003, AAA52040, CAA25879, CAA29886, CAA30229, CAA32583, CAA33387, AAF59902, AAA53474, AAA53475, AAC33527, AAC33528, CAA38276, CAC19501, CAC19502]
17: [GO_0007397, GO_0005201, GO_0007155]; [AAF59902, AAA53474, AAA53475, AAC33527, AAC33528, CAA38276, CAC19501, CAC19502]
18: [GO_0007397, GO_0005201, GO_0005581, GO_0007155, GO_0007517, GO_0005588]; [AAF59902]
19: [GO_0008015, GO_0005201]; [AAA52003, AAA52040, CAA25879, CAA29886, CAA30229, CAA32583, CAA33387, AAA18942, AAA18943, AAA21610, AAA51556, AAA52044, AAB19637, BAA25064, CAA56335, CAC36101]
20: [GO_0008015, GO_0005201, GO_0005581, GO_0007397, GO_0005586]; [AAA52003, AAA52040, CAA25879, CAA29886, CAA30229, CAA32583, CAA33387]
21: [GO_0005587, GO_0005201]; [AAA52006, AAA52042, AAA53098, CAA29075,
CA68698, AAA53097, AAA53099, AAA58422, CAA29076, CAA29098, CAA31275,
AAA18942, AAA18943, AAA21610, AAA51556, AAA52044, AAB19637, BAA25064,
CAA56335, CAC36101, AAA16338, AAA19569, AAB19038, AAB19039, BAA04809,
CAA20120, CAA22265, CAB89263, CAB96748]
22: [GO_0005587, GO_0005201, GO_0007155, GO_0005194]; [AAA18942, AAA18943,
AAA21610, AAA51556, AAA52044, AAB19637, BAA25064, CAA56335, CAC36101,
AAA16338, AAA19569, AAB19038, AAB19039, BAA04809, CAA20120, CAA22265,
CAB89263, CAB96748]
23: [GO_0030198, GO_0005201, GO_0005587]; [AAA53097, AAA53099, AAA58422,
CAA29076, CAA29098, CAA31275, AAA16338, AAA19569, AAB19038, AAB19039,
BAA04809, CAA20120, CAA22265, CAB89263, CAB96748]
24: [GO_0030198, GO_0005201, GO_0007155, GO_0005587, GO_0005194]; [AAA16338,
AAA19569, AAB19038, AAB19039, BAA04809, CAA20120, CAA22265, CAB89263,
CAB96748]
25: [GO_0005194, GO_0007155]; [AAA18942, AAA18943, AAA21610, AAA51556,
AAA52044, AAB19637, BAA25064, CAA56335, CAC36101, AAA16338, AAA19569,
AAB19038, BAA04809, CAA20120, CAA22265, CAB89263, CAB96748,
AAA36524, AAA36786, AAA60266, AAH03072, AAH04257, CAA31408, CAA33333,
CAA33787, CAA39792]
26: [GO_0007517]; [AAF59902, AAC12865, AAC51682, AAC79118, AAH30779, BAA83025]
27: [GO_0000004, GO_0007155]; [AAA53474, AAA53475, AAC33527, AAC33528,
CAA38276, CAA38277, CAC19501, CAC19502, AAA36524, AAA36786, AAA60266,
AAH03072, AAH04257, CAA31408, CAA33333, CAA33787, CAA39792]
28: [GO_0000004, GO_0005201, GO_0007155, GO_0007397, GO_0005587, GO_0005194];
[AAA53474, AAA53475, AAC33527, AAC33528, CAA38276, CAA38277, CAC19501,
CAC19502]
29: [GO_0000004, GO_0007155, GO_0005194, GO_0008372, GO_0008151, GO_0016020,
GO_0004713, GO_0004872, GO_0005524, GO_0016640, GO_0005648, GO_0015021,
GO_0005509, GO_0007156, GO_0007165, GO_0007497, GO_0005554, GO_0004714,
GO_0008603, GO_0005952, GO_0007164]; [AAA36524, AAA36786, AAA60266, AAH03072,
AAH04257, CAA31408, CAA33333, CAA33787, CAA39792]
30: [GO_0005578, GO_0005201, GO_0005594]; [AAA53474, AAA53475, AAC33527,
AAC33528, CAA38276, CAA38277, CAC19501, CAC19502, AAB89077, AAC33525,
CAB81611]
31: [GO_0005578, GO_0005201, GO_0001501, GO_0005594]; [AAA80977, AAC33512,
CAB81611]
32: [GO_0004722, GO_0016787, GO_0006470, GO_0008372]; [AAC79117, AAD40368,
CAA12271, AAC12865, AAC51682, AAC79118, AAH30779, BAA83025, AAC78841,
AAH40012, AAL01037, CAD89918]
33: [GO_0004722, GO_0007517, GO_0016787, GO_0006470, GO_0008372, GO_0004727, GO_0008151, GO_0008138]; [AAC12865, AAC51682, AAC79118, AAH30779, BAA83025]
34: [GO_0004725, GO_0016787, GO_0006470]; [AAC79117, AAD40368, CAA12271, AAC78841, AAH40012, AAL01037, CAD89918, AAC77820]
35: [GO_0004725, GO_0004722, GO_0016787, GO_0006470, GO_0008372]; [AAC79117, AAD40368, CAA12271, AAC78841, AAH40012, AAL01037, CAD89918]
36: [GO_0016787, GO_0006470]; [AAC79117, AAD40368, CAA12271, AAC79118, AAD40368, CAA12271, AAC78841, AAH40012, AAL01037, CAD89918, AAC77820, AAH12399, BAA90964]
37: [GO_0006470]; [AAC79117, AAD40368, CAA12271, AAC79118, AAD40368, CAA12271, AAC78841, AAH40012, AAL01037, CAD89918, AAC77820, AAH12399, BAA90964]
38: [GO_0008372]; [AAC79117, AAD40368, CAA12271, AAC79118, AAD40368, CAA12271, AAC78841, AAH40012, AAL01037, CAD89918, AAA36524, AAA36786, AAA60266, AAH03072, AAH04257, CAA31408, CAA33333, CAA33787, CAA39792]
39: [GO_0004727, GO_0016787, GO_0006470, GO_0008138]; [AAC12865, AAC51682, AAC79118, AAD40368, CAA12271, AAC79118, AAD40368, CAA12271, AAC78841, AAH40012, AAL01037, CAD89918, AAC77820, AAH12399, BAA90964]
40: [GO_0008151, GO_0008372]; [AAC12865, AAC51682, AAC79118, AAD40368, CAA12271, AAC79118, AAD40368, CAA12271, AAC78841, AAH40012, AAL01037, CAD89918, AAA36524, AAA36786, AAA60266, AAH03072, AAH04257, CAA31408, CAA33333, CAA33787, CAA39792]
41: [GO_0005737, GO_0016787, GO_0006470, GO_0004727, GO_0008138, GO_0016020]; [AAC79119, AAF40203, AAF40204, AAF40205, BAA20826]
42: [GO_0016020]; [AAC79119, AAF40203, AAF40204, AAF40205, BAA20826, AAA36524, AAA36786, AAA60266, AAH03072, AAH04257, CAA31408, CAA33333, CAA33787, CAA39792]
43: [GO_0004721, GO_0006470]; [AAC35609, AAH12399, BAA90964]
44: [GO_0004721, GO_0016787, GO_0006470]; [AAH12399, BAA90964]
45: [GO_0008270, GO_0006470, GO_0004721]; [AAH35609]
46: [GO_0006118, GO_0004725, GO_0016787, GO_0006470]; [AAC77820]
47: [GO_0004713, GO_0005524, GO_0006468]; [AAA35835, AAA35836, AAA35837, AAA35840, AAA35958, AAA35959, AAA35960, AAA75007, CAA01958, CAA36101, CAA37015, CAA40400, CAA40401, CAA40402, CAA40403, CAA40404, CAA47375, CAA68679, AAA36147, AAA36152, AAA52449, AAA59470, AAA59471, AAA59472, AAA61188, AAA68514, AAC41933, AAC41934, AAC41935, AAC41936, AAC41937, AAC1938, AAD14392, AAD31560, AAD31561, AAD31562, AAD31565, AAD31567, AAF43273, AAF43274, AAK94205, AAK94206, AAK94207, AAK94208, AAK94209, AAM74056, BAA89296, BAA89297, BAA89298, BAA89299, BAA89300, BAA89301,
BAC45037, CAA37014, CAA39654, CAA76643, CAA96492, AAA52450, AAA58470, AAA63209, AAA67781, AAB33323, AAF63380, AAK54727, AAK54729, CAA59334, AAA63208, AAB59389, AAA11847, AAK34949, CAA40490, CAA74200, CAA89250, AAA36524, AAA36786, AAA60266, AAH03072, AAH04257, CAA31408, CAA33333, CAA33787, CAA39792]

48: [GO_0004713, GO_0001501, GO_0004872, GO_0005007, GO_0005524, GO_0016740, GO_0006468, GO_0000165, GO_0008543, GO_0005887]; [AAA35835, AAA35836, AAA35837, AAA35840, AAA35958, AAA35959, AAA35960, CAA75007, AAA1958, AAA63101, CAA40400, CAA40401, CAA40402, CAA40403, CAA47375, CAA68679, AAA36147, AAA36152, AAA52449, AAA59470, AAA59471, AAA59472, AAA61188, AAA68514, AAC41933, AAC41934, AAC41935, AAC41936, AAC41937, AAC41938, AAD14392, AAD31560, AAD31561, AAD31562, AAD31565, AAD31567, AAF43273, AAF43274, AAK94205, AAK94206, AAK94207, AAK94208, AAM74056, BAA89296, BAA89297, BAA89298, BAA89300, BAA89301, CAA37014, CAA39654, CAA76643, CAA96492, AAA52450, AAA58470, AAA63209, AAA67781, AAB33323, AAF63380, AAK54727, AAK54729, CAA59334]

49: [GO_0004872]; [AAA35835, AAA35836, AAA35837, AAA35838, AAA35839, AAA35840, AAA35958, AAA35959, AAA35960, AAA75007, AAA36101, CAA40400, CAA40401, CAA40402, CAA40403, CAA47375, CAA68679, AAA36147, AAA36152, AAA52449, AAA59470, AAA59471, AAA59472, AAA61188, AAA68514, AAC41933, AAC41934, AAC41935, AAC41936, AAC41937, AAC41938, AAD14392, AAD31560, AAD31561, AAD31562, AAD31565, AAD31567, AAF43273, AAF43274, AAK94205, AAK94206, AAK94207, AAK94208, AAM74056, BAA89296, BAA89297, BAA89298, BAA89300, BAA89301, CAA37014, CAA39654, CAA76643, CAA96492, AAA52450, AAA58470, AAA63209, AAA67781, AAB33323, AAF63380, AAK54727, AAK54729, CAA59334, AAA61139, BAC45250, CAA43290]

50: [GO_0004872, GO_0005198, GO_0005524, GO_0016740, GO_0006468, GO_0005887, GO_0007165, GO_0004714]; [AAA61139, CAA43290]

51: [GO_0004872, GO_0004713, GO_0005524, GO_0016740, GO_0006468]; [AAA35835, AAA35836, AAA35837, AAA35840, AAA35958, AAA35959, AAA35960, AAA75007, AAA1958, CAA36101, CAA37015, CAA40400, CAA40401, CAA40402, CAA40403, CAA40404, CAA47375, CAA68679, AAA36147, AAA36152, AAA52449, AAA59471, AAA59472, AAA61188, AAA68514, AAC41933, AAC41934, AAC41935, AAC41936, AAC41937, AAC41938, AAD14392, AAD31560, AAD31561, AAD31562, AAD31565, AAD31567, AAF43273, AAF43274, AAK94205, AAK94206, AAK94207, AAK94208, AAM74056, BAA89296, BAA89297, BAA89298, BAA89300, BAA89301, CAA37014, CAA39654, CAA76643, CAA96492, AAA52450, AAA58470, AAA63209, AAA67781, AAB33323, AAF63380, AAK54727, AAK54729, CAA59334, AAA63208, AAA63209, AAA36786, AAA60266, AAH03072, AAH04257, CAA31408, CAA33333, CAA33787, CAA39792, AAA61139, BAC45250, CAA43290]

52: [GO_0005007, GO_0004872]; [AAA35835, AAA35836, AAA35837, AAA35838, AAA35839, AAA35840, AAA35958, AAA35959, AAA35960, AAA75007, AAA1958, AAA63209, AAA67781, AAB33323, AAF63380, AAK54727, AAK54729, CAA59334, AAA36524, AAA36786, AAA60266, AAH03072, AAH04257, CAA31408, CAA33333, CAA33787, CAA39792]
CAA36101, CAA37015, CAA40400, CAA40401, CAA40402, CAA40403, CAA40404, CAA47375, CAA68679, AAA36147, AAA36152, AAA52449, AAA59470, AAA59471, AAA59472, AAA61188, AAA68514, AAC41933, AAC41934, AAC41935, AAC41936, AAC41937, AAC41938, AAD14392, AAD31560, AAD31561, AAD31562, AAD31565, AAD31567, AAF43273, AAF43274, AAK94205, AAK94206, AAK94207, AAK94208, AAK94209, AAM74056, BAA89296, BAA89297, BAA89298, BAA89299, BAA89300, BAA89301, CAA37014, CAA39654, CAA76643, CAA96492, AAA52450, AAA58470, AAA63209, AAA67781, AAB33323, AAF63380, AAK54727, AAK54729, CAA59334, AAA63208, AAB59389, AAH11847, CAA40490, CAA74200] 53: [GO_0005007, GO_0004713, GO_0004872, GO_0016740, GO_0006468]; [AAA35835, AAA35836, AAA35837, AAA35840, AAA35958, AAA35959, AAA35960, AAA75007, CAA01958, CAA36101, CAA37015, CAA40400, CAA40401, CAA40402, CAA40403, CAA40404, CAA47375, CAA68679, AAA36147, AAA36152, AAA52449, AAA59470, AAA59471, AAA59472, AAA61188, AAA68514, AAC41933, AAC41934, AAC41935, AAC41936, AAC41937, AAC41938, AAD14392, AAD31560, AAD31561, AAD31562, AAD31565, AAD31567, AAF43273, AAF43274, AAK94205, AAK94206, AAK94207, AAK94208, AAK94209, AAM74056, BAA89296, BAA89297, BAA89298, BAA89299, BAA89300, BAA89301, CAA37014, CAA39654, CAA76643, CAA96492, AAA52450, AAA58470, AAA63209, AAA67781, AAB33323, AAF63380, AAK54727, AAK54729, CAA59334, AAA63208, AAB59389, AAH11847, CAA40490, CAA74200] 54: [GO_0005007, GO_0006468]; [AAA35835, AAA35836, AAA35837, AAA35840, AAA35958, AAA35959, AAA35960, AAA75007, CAA01958, CAA36101, CAA37015, CAA40400, CAA40401, CAA40402, CAA40403, CAA47375, CAA68679, AAA36147, AAA36152, AAA52449, AAA59470, AAA59471, AAA59472, AAA61188, AAA68514, AAC41933, AAC41934, AAC41935, AAC41936, AAC41937, AAC41938, AAD14392, AAD31560, AAD31561, AAD31562, AAD31565, AAD31567, AAF43273, AAF43274, AAK94205, AAK94206, AAK94207, AAK94208, AAK94209, AAM74056, BAA89296, BAA89297, BAA89298, BAA89299, BAA89300, BAA89301, CAA37014, CAA39654, CAA76643, CAA96492, AAA52450, AAA58470, AAA63209, AAA67781, AAB33323, AAF63380, AAK54727, AAK54729, CAA59334, AAA63208, AAB59389, AAH11847, CAA40490, CAA74200, CAB89250, AAA36524, AAA36786, AAA60266, AAH03072, AAH04257, CAA31408, CAA33333, CAA33787, CAA39792, AAA61139, AAA43290] 55: [GO_0005007, GO_0004713, GO_0004872, GO_0016740, GO_0006468]; [AAA35835, AAA35836, AAA35837, AAA35840, AAA35958, AAA35959, AAA35960, AAA75007, CAA01958, CAA36101, CAA37015, CAA40400, CAA40401, CAA40402, CAA40403, CAA47375, CAA68679, AAA36147, AAA36152, AAA52449, AAA59470, AAA59471, AAA59472, AAA61188, AAA68514, AAC41933, AAC41934, AAC41935, AAC41936, AAC41937, AAC41938, AAD14392, AAD31560, AAD31561, AAD31562, AAD31565, AAD31567, AAF43273, AAF43274, AAK94205, AAK94206, AAK94207, AAK94208, AAK94209, AAM74056, BAA89296, BAA89297, BAA89298, BAA89299, BAA89300, BAA89301, CAA37014, CAA39654, CAA76643, CAA96492, AAA52450, AAA58470, AAA63209, AAA67781, AAB33323, AAF63380, AAK54727, AAK54729, CAA59334, AAA63208, AAB59389, AAH11847, AAK34949, CAA40490, CAA74200, CAA31408, CAA33333, CAA33787, CAA39792, AAA61139, AAA43290]

133
BAA89301, CAA37014, CAA39654, CAA76643, CAA96492, AAA52450, AAA58470, AAA63209, AAA67781, AAB33323, AAF63380, AAK54727, AAK54729, CAA9334, AAA63208, AAB59389, AAH11847, AAK34949, CAA40490, CAA74200, CAA36524, AAA36786, AAA60266, AAH03072, AAH04257, CAA31408, CAA33333, CAA33787, CAA39792, AAA61139, CAA43290]
56: [GO_0008201, GO_0001501, GO_0005524, GO_0016740, GO_0006468, GO_0000165, GO_0008543, GO_0005887]; [AAA35835, AAA35836, AAA35837, AAA35840, AAA35958, AAA35960, AAA75007, CAA01958, CAA36101, CAA40400, CAA40401, CAA40402, CAA40403, AAA36147, AAA36152, AAA52449, AAA59470, AAA59471, AAA59472, AAA61188, AAA68514, AAC41933, AAC41934, AAC41935, AAC41936, AAC41937, AAC41938, AAD14392, AAD31560, AAD31561, AAD31562, AAD31565, AAD31567, AAF43273, AAF43274, AAK94205, AAK94206, AAK94207, AAK94208, AAK94209, AAM74056, BAA89296, BAA89297, BAA89298, BAA89299, BAA89300, BAA89301, CAA37014, CAA39654, CAA76643, CAA96492, AAA36524, AAA36786, AAA60266, AAH03072, AAH04257, CAA31408, CAA33333, CAA33787, CAA39792]
57: [GO_0016021, GO_0004872]; [AAA35835, AAA35836, AAA35837, AAA35838, AAA35839, AAA35840, AAA35958, AAA35960, AAA75007, CAA01958, CAA36101, CAA40400, CAA40401, CAA40402, CAA40403, AAA36147, AAA36152, AAA52449, AAA59470, AAA59471, AAA59472, AAA61188, AAA68514, AAC41933, AAC41934, AAC41935, AAC41936, AAC41937, AAC41938, AAD14392, AAD31560, AAD31561, AAD31562, AAD31565, AAD31567, AAF43273, AAF43274, AAK94205, AAK94206, AAK94207, AAK94208, AAK94209, AAM74056, BAA89296, BAA89297, BAA89298, BAA89299, BAA89300, BAA89301, CAA37014, CAA39654, CAA76643, CAA96492, AAA36524, AAA36786, AAA60266, AAH03072, AAH04257, CAA31408, CAA33333, CAA33787, CAA39792]
58: [GO_0016021, GO_0004872, GO_0004713, GO_0005524, GO_0006468, GO_0004872, GO_0016740]; [AAA35835, AAA35836, AAA35837, AAA35838, AAA35839, AAA35840, AAA35958, AAA35960, AAA75007, CAA01958, CAA36101, CAA40400, CAA40401, CAA40402, CAA40403, AAA36147, AAA36152, AAA52449, AAA59470, AAA59471, AAA59472, AAA61188, AAA68514, AAC41933, AAC41934, AAC41935, AAC41936, AAC41937, AAC41938, AAD14392, AAD31560, AAD31561, AAD31562, AAD31565, AAD31567, AAF43273, AAF43274, AAK94205, AAK94206, AAK94207, AAK94208, AAK94209, AAM74056, BAA89296, BAA89297, BAA89298, BAA89299, BAA89300, BAA89301, CAA37014, CAA39654, CAA76643, CAA96492, AAA36524, AAA36786, AAA60266, AAH03072, AAH04257, CAA31408, CAA33333, CAA33787, CAA39792]
60: \{GO_0016021, GO_0004713, GO_0005524, GO_0006468, GO_0004872, GO_0016740, GO_0005007\}; \{AAA35835, AAA35836, AAA35837, AAA35840, AAA35958, AAA35959, AAA35960, AAA75007, CAA01958, CAA36101, CAA37015, CAA40400, CAA40401, CAA40402, CAA40403, CAA40404, CAA47375, CAA68679, AAA36147, AAA36152, AAA52449, AAA59470, AAA59471, AAA59472, AAA61188, AAA68514, AAC41933, AAC41934, AAC41935, AAC41936, AAC41937, AAC41938, AAD14392, AAD31560, AAD31561, AAD31562, AAD31565, AAD31567, AAF43273, AAF43274, AAK94205, AAK94206, AAK94207, AAK94208, AAK94209, AAM74056, BAA89296, BAA89297, BAA89298, BAA89299, BAA89300, BAA89301, CAA37014, CAA39654, CAA76643, CAA96492\}

61: \{GO_0008543, GO_0004713, GO_0005524, GO_0006468, GO_0004872, GO_0016740, GO_0005007, GO_0005887\}; \{AAA35835, AAA35836, AAA35837, AAA35958, AAA35959, AAA35960, AAA75007, CAA01958, CAA36101, CAA37015, CAA40400, CAA40401, CAA40402, CAA40403, CAA40404, CAA47375, CAA68679, AAA52450, AAA58470, AAA63209, AAA67781, AAB33323, AAF63380, AAK54727, AAK54729, CAA59334, AAA63208, AAB59389, AAH11847, CAA40490, CAA74200\}

62: \{GO_0005887, GO_0004872\}; \{AAA35835, AAA35836, AAA35837, AAA35958, AAA35959, AAA35960, AAA75007, CAA01958, CAA36101, CAA37015, CAA40400, CAA40401, CAA40402, CAA40403, CAA40404, CAA47375, CAA68679, AAA52450, AAA58470, AAA63209, AAA67781, AAB33323, AAF63380, AAK54727, AAK54729, CAA59334, AAA63208, AAB59389, AAH11847, CAA40490, CAA74200, AAA61139, BAC45250, CAA43290\}

63: \{GO_0005887, GO_0004872, GO_0005524, GO_0006468, GO_0016740\}; \{AAA35835, AAA35836, AAA35837, AAA35958, AAA35959, AAA35960, AAA75007, CAA01958, CAA36101, CAA37015, CAA40400, CAA40401, CAA40402, CAA40403, CAA40404, CAA47375, CAA68679, AAA52450, AAA58470, AAA63209, AAA67781, AAB33323, AAF63380, AAK54727, AAK54729, CAA59334, AAA63208, AAB59389, AAH11847, CAA40490, CAA74200, AAA61139, BAC45250, CAA43290\}

64: \{GO_0005887, GO_0004872, GO_0005007\}; \{AAA35835, AAA35836, AAA35837, AAA35958, AAA35959, AAA35960, AAA75007, CAA01958, CAA36101, CAA37015, CAA40400, CAA40401, CAA40402, CAA40403, CAA40404, CAA47375, CAA68679, AAA52450, AAA58470, AAA63209, AAA67781, AAB33323, AAF63380, AAK54727, AAK54729, CAA59334\}

65: \{GO_0016049, GO_0001501, GO_0004713, GO_0005524, GO_0006468, GO_0004872, GO_0005007, GO_0016740, GO_0000165, GO_0008543, GO_0005887, GO_0007259\}; \{AAA52450, AAA58470, AAA63209, AAA67781, AAB33323, AAF63380, AAK54727, AAK54729, CAA59334\}

66: \{GO_0016049, GO_0004872, GO_0005007, GO_0016021, GO_0005624\}; \{AAA35835, AAA35836, AAA35837, AAA35840, AAA35958, AAA35959, AAA35960, AAA75007, CAA01958, CAA36101, CAA37015, CAA40400, CAA40401, CAA40402, CAA40403, CAA40404, CAA47375, CAA68679, AAA52450, AAA58470, AAA63209, AAA67781, AAB33323, AAF63380, AAK54727, AAK54729, CAA59334\}

67: \{GO_0004674, GO_0004713, GO_0005524, GO_0006468\}; \{BAC45037\}

68: \{GO_0007165, GO_0004872, GO_0004714\}; \{AAA36524, AAA36786, AAA60266, AAH03072, AAH04257, CAA31408, CAA33333, CAA33787, CAA39792, AAA61139, BAC45250, CAA43290\}
69: [GO_0007165, GO_0004872, GO_0005524, GO_0006468, GO_0016740, GO_0004714]; [AAA36524, AAA36786, AAA60266, AAH03072, AAH04257, CAA31408, CAA33333, CAA33787, CAA39792, AAA61139, CAA43290]
70: [GO_0007165, GO_0004872, GO_0005887, GO_0004714]; [AAA61139, BAC45250, CAA43290]
71: [GO_0007169, GO_0004872, GO_0005887, GO_0007165, GO_0004714, GO_0007267]; [AAA61139, BAC45250]
72: [GO_0007169, GO_0005198, GO_0004872, GO_0005524, GO_0016740, GO_0006468, GO_0005887, GO_0007165, GO_0004714, GO_0007267]; [AAA61139]
73: [GO_0007498, GO_0005198, GO_0004872, GO_0005524, GO_0016740, GO_0006468, GO_0005887, GO_0007165, GO_0004714]; [CAA43290]

Appendix: concept pairs that have high similarity greater than 0.9 in breast cancer
example in section 7.4

training concept #4
Attributes: [GO_0005634]
Objects: [472, 580, 672, 675, 5888, 7157, 8493, 9821, 11200, 79728, 83990]

test concept #4
Attributes: [GO_0005634]
Objects: [1488, 2018, 3167, 4288, 9184, 10579, 11023, 11101, 26098, 54764, 56647, 59338]
Similarity: 1.0

training concept #7
Attributes: [GO_0007049, GO_0005634]
Objects: [472, 672, 7157, 8493, 9821, 11200]

test concept #16
Attributes: [GO_0007049, GO_0005634]
Objects: [4288, 56647]
Similarity: 1.0

training concept #10
Attributes: [GO_0016740]
Objects: [472, 5290, 11200]

test concept #59
Attributes: [GO_0016740]
Objects: [2263, 2869, 4942, 11101, 51363, 118672]
Similarity: 1.0

training concept #11
Attributes: [GO_0045786, GO_0005634, GO_0007049]
Objects: [472, 672, 7157, 9821]

training concept #16
Attributes: [GO_0007049, GO_0005634]
Objects: [4288, 56647]
Similarity: 0.9147689938545227

training concept #17
Attributes: [GO_0005515]
Objects: [580, 672, 675, 841, 3845, 5888, 7157, 9821, 11200, 83990]

training concept #20
Attributes: [GO_0005515, GO_0005634]
Objects: [580, 672, 675, 5888, 7157, 9821, 11200, 83990]

training concept #29
Attributes: [GO_0005737, GO_0005634, GO_0005515]
Objects: [580, 675, 7157, 83990]

Similarity: 0.9076853394508362