Available online at www.sciencedirect.com
ScienceDirect
Journal of Biomedical Informatics 41 (2008) 65–81
Journal of Biomedical Informatics
www.elsevier.com/locate/yjbin

A genetic similarity algorithm for searching the Gene Ontology terms and annotating anonymous protein sequences

Razib M. Othman [a,*], Safaai Deris [a], Rosli M. Illias [b]

[a] Department of Software Engineering, Faculty of Computer Science and Information Systems, Universiti Teknologi Malaysia, 81310 UTM Skudai, Malaysia
[b] Department of Bioprocess, Faculty of Chemical and Natural Resources Engineering, Universiti Teknologi Malaysia, 81310 UTM Skudai, Malaysia

Received 22 September 2006
Available online 27 June 2007

Abstract

A genetic similarity algorithm is introduced in this study to find a group of semantically similar Gene Ontology terms. The genetic similarity algorithm combines semantic similarity measure algorithm with parallel genetic algorithm. The semantic similarity measure algorithm is used to compute the similitude strength between the Gene Ontology terms. Then, the parallel genetic algorithm is employed to perform batch retrieval and to accelerate the search in large search space of the Gene Ontology graph. The genetic similarity algorithm is implemented in the Gene Ontology browser named *basic* UTMGO to overcome the weaknesses of the existing Gene Ontology browsers which use a conventional approach based on keyword matching. To show the applicability of the *basic* UTMGO, we extend its structure to develop a Gene Ontology -based protein sequence annotation tool named *extended* UTMGO. The objective of developing the *extended* UTMGO is to provide a simple and practical tool that is capable of producing better results and requires a reasonable amount of running time with low computing cost specifically for offline usage. The computational results and comparison with other related tools are presented to show the effectiveness of the proposed algorithm and tools.

Keywords: Gene Ontology; Genetic algorithm; Semantic similarity measure; Genetic similarity search; Protein sequence annotation

1. Introduction

As outlined by the European Bioinformatics Institute (EBI), annotation of an anonymous protein sequence should be inferred from annotations of the nucleotide sequences, analogies with already understood proteins, plus references to patterns and motifs as characteristics of particular protein functions. Annotation of anonymous protein sequences is important for the preservation and reuse of knowledge and for content-based queries. Traditional wet-lab methods are labor intensive and prone to human error. On the other hand, sequence-similarity-based tools like Basic Local Alignment Search Tool (BLAST) are time intensive and require high investment in computing facilities such as cluster server or grid computing if being used locally. Furthermore, for remote users, these tools are subject to internet stability and speed to access the tools and to get the results online. Therefore, a simple and practical method that is capable of producing better results and requires a reasonable amount of running time with low computing cost specifically for offline usage is needed.

In the last few years, the Gene Ontology (GO) [1] terms have been widely used to annotate various protein sets such as in NOPdb [2], a database of nucleolar proteome; SCOPPI [3], a database of protein domain–domain interactions; DRTF [4], a database of rice transcription factor; and MolMovDB [5], a database of macromolecular motions. In addition, GO terms have been successfully implemented in large-scale protein annotation projects involving SWISS-PROT, TrEMBL, and InterPro databases [6]. The GO is a collection of nearly 23 thousand terms to describe gene and gene product attributes in any organism. The GO terms are structured, controlled vocabularies organized as

* Corresponding author. Fax: +607 5565044.
E-mail address: razib@utm.my (R.M. Othman).

1532-0464/$ - see front matter © 2007 Elsevier Inc. All rights reserved.
doi:10.1016/j.jbi.2007.05.010

a Directed Acyclic Graph (DAG) in three aspects: cellular component, biological process, and molecular function. Let GO be a graph $G = \{V, E\}$, where V is a set of nodes representing the GO terms and E is a set of pairs of nodes representing relationships between the GO terms. The GO terms can have more than one parent, as well as multiple children. The GO terms are connected by two relationships: the "is-a" relationship, e.g. "chromatin binding" (GO:0003682) and "structure-specific DNA binding" (GO:0043566) are parents of "chromatin DNA binding" (GO:0031490); and the "part-of" relationship, e.g. "cytoplasmic part" (GO:0044444) is part of "cytoplasm" (GO:0005737). The advantages of using the GO are as follows: the GO data is dynamic and constantly evolves according to the advances in current state of biological knowledge; the GO data is publicly available and can be downloaded at any time from the World Wide Web (WWW) in MySQL, RDF/XML, OBO/XML, and OWL formats that can be understandable and processable by human and machine alike; the common GO terms shared by gene and protein sequences in multiple organisms in different databases can facilitate uniform queries across them; and the association of GO terms with nearly 2.5 million gene products supported by the evidence and citation can affirm its reliability for future evaluation and use. The link between the GO terms and gene products is provided by the Gene Ontology Annotation (GOA) [7]. In the GOA project, electronic mappings and manual curation are used to assign the GO terms to all proteomes existing in the UniProt, Ensembl, and other organism databases. It covers 2.3 million protein sequences from 0.26 million species.

However, application of the GO terms to annotate anonymous protein sequences is not easy, especially for species not yet inserted in public biological databases. Furthermore, for bioscientists with little computational knowledge or limited facilities it is a hard task to annotate those anonymous protein sequences. The difficulties arise because generally the existing GO-based tools are (1) dependent on BLAST which is computationally intensive and requires high-cost and high-specification hardware since sequence alignment is performed to all protein sequences but not only to protein sequences that indicate higher similarity, (2) dependent on Relational Database Management Systems (RDBMS) which require the user to setup the RDBMS software and to import the data or sources into the RDBMS format, and (3) partially based on the GO data which requires the user to download the GOA data or protein sequence data sets from several sources.

Therefore, in this study, a new way of applying the GO terms to annotate anonymous protein sequences is introduced. The method consists of three main components. In the first component, the single monolithic GO RDF/XML file is split into smaller files. It is carried out to avoid dependency on RDBMS format, to provide all-in-one source by adding protein sequences and Inferred from Electronic Annotation (IEA) evidence associations into the files since they are not included in the original GO RD/XML file, and to make the GO data easily accessible and processable. In the second component, the main focus of this paper, semantic similarity search is performed over the smaller GO RDF/XML files. The target is to find a group of semantically similar GO terms with higher term similarity score to a GO term which is foreseen to have higher relationship with the query protein sequence. Lastly, the results obtained from the second component are verified by computing sequence alignment score between the query protein sequence and all protein sequences attached to those GO terms. With this method, sequence alignment is carried out only to protein sequences with higher outguessed similarity. Hence, demand for high computational facilities and execution time can be reduced. A GO-based tool named *extended* UTMGO is developed to demonstrate the method. The *extended* UTMGO employs a GO browser named *basic* UTMGO for implementing the second component. The JAligner engine (http://jaligner.sourceforge.net) that uses the Smith–Waterman algorithm has been integrated and modified to perform the sequence alignment and to comply with the *extended* UTMGO. The flow of the *extended* UTMGO can be summarized as shown in Fig. 1.

The semantic similarity searching is related to the problem of determining semantic relatedness between terms either by virtue of their likeness (*bank–trust company*), synonymy (*car–automobile*), meronymy (*computer–keyboard*), antonymy (*rich–poor*), functional relationship (*marker pen–white board*), or frequent association (*orang utan–Borneo*). For finding semantically similar GO terms, the terms are related according to the association: a table that stores

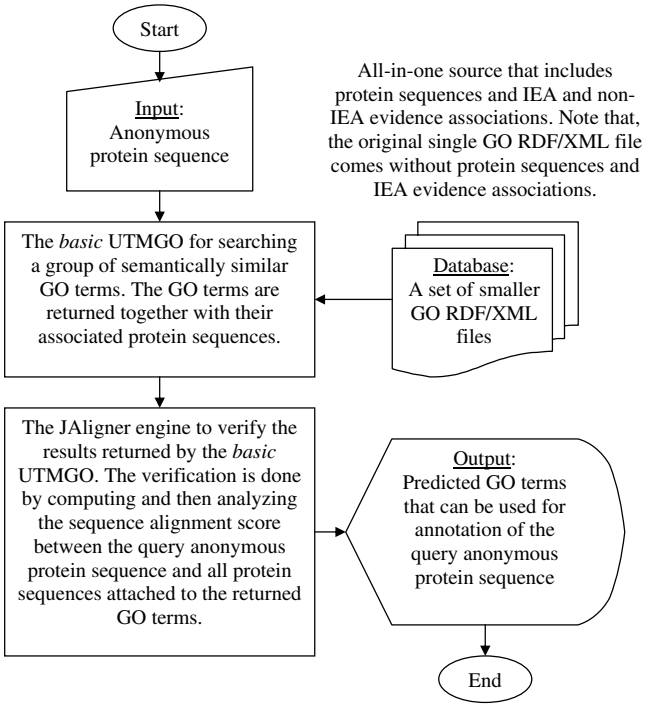

Fig. 1. The flowchart of the *extended* UTMGO.

information shared among the GO terms. Particularly, this table provides an annotation record that is basically a link between a gene product and a GO term provided by the GOA. However, the existing GO browsers that support basic needs of bioscientists for searching the GO terms still use a conventional approach based on keyword matching. Thus, for bioscientists, finding a group of semantically similar GO terms is time consuming and a tedious task. For example, the keyword matching is not capable of computing the relationship between "intracellular organelle" (GO:0043229) and "cytoplasm" (GO:0005737) even though they share the same parent "intracellular part" (GO:0044424) because their names do not exactly or approximately match. Therefore, the *basic* UTMGO uses a genetic similarity algorithm that incorporates the parallel genetic algorithm and the semantic similarity measure algorithm. The parallel genetic algorithm is used to generate a solution consisting of a group of semantically similar GO terms that best match to the query GO term, and to accelerate the search in the large GO graph. The search space of the GO graph, $g(k)$, is astronomical and varies between:

$$2^{\frac{k(k-1)}{2}} \leq g(k) \leq 3^{\frac{k(k-1)}{2}}, \quad (1)$$

where k is the number of nodes in the GO graph. Currently the GO graph consists of 22,954 nodes, so the search space of the GO graph is between $2^{263,431,581}$ and $3^{263,431,581}$. A parallel genetic algorithm optimizes its fitness function by utilizing the genetic operators to find an optimal solution. It can also be executed on a low-cost PC cluster using message passing interface libraries that are open source and easy to install. The semantic similarity measure algorithm is added into the parallel genetic algorithm to measure the similitude strength between the GO terms during the creation of initial population and calculation of fitness value. The semantic similarity measure algorithm used is a combination of information content (node-based) and conceptual distance (edge-based). The information content is used to get the amount of information the GO terms share in common, whereas the conceptual distance is applied to know the depth and the local network density of the GO terms.

The remainder of the paper consists of related work in semantic similarity measures and genetic algorithms and existing tools for searching the GO terms and for annotating anonymous protein sequences (Section 2), detailed explanation of the proposed genetic similarity algorithm (Section 3), description of the computational environment and data used in this study including step-by-step explanations of both the basic and extended versions of UTMGO (Section 4), the results and discussion of experiments (Section 5), and the general conclusions (Section 6).

2. Related work

Semantic similarity measures play an important role in information retrieval and natural language processing. Example applications include characterization of human regulatory pathways [8], linguistic modeling [9], computer-assisted inter-observer consensus [10], and semantic feature ratings [11]. The choice of semantic similarity measure has the ability to improve the recall and precision of information retrieval by identifying the relation between concepts. This is done by calculating the distance or the amount of information in common between the two concepts being analyzed. Most of the popular measures are based on taxonomic or ontological structure [12–15]. These measures have been analyzed by Budanitsky and Hirst [16], and the evaluation of WordNet (http://wordnet.princeton.edu) based semantic similarity measures in their study shows that the Jiang and Conrath semantic similarity measure [14] provides the best results. The Jiang and Conrath semantic similarity measure is a combined approach that inherits the conceptual distance approach enhanced with the information content approach. The basic calculation of the Jiang and Conrath semantic similarity measure is expressed as:

$$dist(c_1, c_2) = IC(c_1) + IC(c_2) - 2 \times sim(c_1, c_2), \quad (2)$$

where $IC(c) = -\log P(c)$, $sim(c_1, c_2) = \max_{c \in S(c_1, c_2)} \{IC(c)\}$, c is some concept being studied, $P(c)$ is the probability of encountering an instance of concept c, and $S(c_1, c_2)$ is the set of concepts that subsume both c_1 and c_2.

Lord et al. [17] has studied the Resnik [15] semantic similarity measure on the GO. They have only considered the GO annotations in Swiss-PROT and the "is-a" relationship. Their work has been extended by Popescu et al. [18]. In the meantime, Sevilla et al. [19] has compared different semantic similarity measures proposed by Lin [13], Jiang and Conrath, and Resnik. They conclude that the Resnik semantic similarity measure outperforms the other semantic similarity measures. However, their comparisons are based on the gene products rather than the GO terms, and they used the subsets of the GO terms and annotations. Therefore, in this study we use the Jiang and Conrath semantic similarity measure to compute the semantic similarity between pairs of GO terms rather than between pairs of gene products, and we use all the GO terms and annotations provided by the GO Consortium including the "part-of" relationships. The Jiang and Conrath semantic similarity measure is selected since both notions of the shared information content and the conceptual distance of the GO terms in the GO graph are considered as discussed in Section 3.1.

A genetic algorithm is selected because its capabilities as a machine learning technique have been recognized in the information retrieval field. This is due to its capability of being adaptive, efficient, robust, and a global search method that is suitable to address a situation where the search space is large. The properties of the genetic algorithm are as follows: a chromosome (a string of symbols called genes) to represent a solution, an allele to represent the value of the gene (it is usually a binary

bit {0, 1}, an integer, or a real number), loci to represent the positions of the genes in the chromosome, a population to represent a set of chromosomes, a fitness function to evaluate each chromosome, a set of genetic operators to generate a new population, and a selection method to select fitter chromosomes for the next generation. The genetic algorithm starts with an initialization step in which an initial population is generated at random. Then it evolves with the following steps in each generation: evaluation of fitness function (the value of each chromosome in the population is calculated according to the fitness function), selection (multiple chromosomes are stochastically selected from the current population based on their fitness to form a new population), and a genetic operation (modification is performed to a newly generated population). These steps are repeated until either a maximum number of generations have been produced or a satisfactory fitness level has been reached for the population. Some reviews of genetic algorithms can be found in [20–22]. Implementations of the genetic algorithm in information retrieval are normally related to web search [23], gene selection [24], spatial information retrieval [25], and document retrieval [26].

For searching the GO terms, most of the present GO browsers respond to user queries by retrieving relevant GO terms based on keyword matching. A list of tools for searching and browsing the GO terms can be found at http://www.geneontology.org/GO.tools.browsers.shtml. Among the popular GO browsers are:

- AmiGO is a GO browser developed by the GO Consortium. The keyword-based search is executed either by "exact" or "contains" match over the GO term accession number, name, or synonyms. This tool also allows a user to use a gene product or a protein sequence as a search input.
- GenNav is a GO browser that uses string matching method namely "exact" or "approximate" match that responds to a given GO term or gene product. GenNav is maintained by the United States National Library of Medicine (US NLM).
- QuickGO is a GO browser that allows a user to retrieve the GO terms by "exact" or "wildcard" search for the GO term accession number, name, synonyms, definitions, or comments. This web-based GO browser can be found at the website of the EBI.
- TAIR Keyword Browser is a GO browser that uses the GO term accession number or name as an input and then performs either "contains", "start with", "end with", or "exact" match. This tool is developed by the Arabidopsis Information Resource (TAIR).

Moreover, DynGO [27] and FuSSiMeG [28] are recently developed GO browsers that perform the semantic similarity search over the GO terms. However, the DynGO has only focused on the information content and has overlooked the role of conceptual distance in finding the significant GO terms. Whereas, the FuSSiMeG is not capable of returning more than one GO term for each query.

Several tools have been developed in recent years to annotate anonymous protein sequences in accordance with the GO terms. The generally used tools include:

- GoFigure [29] is a tool that accepts an unknown DNA or protein sequence as an input and then uses BLAST to predict the GO terms by identifying homologous sequences in the GO annotated databases.
- GOtcha [30] is a tool that provides a prediction of a set of GO terms for a given query sequence (DNA or protein). BLAST is used to get the initial score of each GO term and the scores are calibrated against term-specific probability (*P*-score) to give higher accuracy.
- GOPET [31] is an automated annotation tool for assigning the GO terms to cDNA or protein query sequences. It uses BLAST to perform homology searches against GO-mapped protein databases, and support vector machines for the prediction and the assignment of confidence values.
- JAFA [32] is a meta-server that uses several function prediction programs such as GoFigure, GOtcha, GOblet [33], Phydbac [34], and InterProScan [35]. It accepts a protein sequence and returns the predicted GO terms with prediction score that is based on the ratio of agreeing servers.

However, as mentioned earlier in the previous section, for offline usage, these tools are difficult to configure and use, especially by bioscientists. The tools also require an expensive high performance computing environment. Whereas, for online usage, they depend on internet stability and speed.

3. Method

3.1. Semantic similarity measure algorithm

The semantic similarity measure algorithm, as shown in Fig. 2, takes as input a set of subgraphs of the GO graph and the query GO term. It returns a set of subgraphs of the GO graph with assigned term similarity score for each node in the subgraphs. The term similarity score is used for generation of the initial population and evaluation of the fitness function. The semantic similarity measure algorithm described in this section is adopted from the Jiang and Conrath. It is simplified, and a direct explanation of how the GO is applied to their semantic similarity measure is given.

3.1.1. Information content approach

The information content is computed according to the association: a source that presents information shared among the GO terms. The association is a table that stores annotations which provide links between GO terms and gene products that are supported by evidence codes and lit-

```
1   Semantic-Similarity-Measure-Algorithm (G, q);
2   Input: G = {G1, G2,...,Gm} (a set of subgraphs of the GO graph) and q (a
3   query GO term)
4   Output: G' = {G1', G2',... ,Gm'} (a set of subgraphs of the GO graph with
5   assigned term similarity score)
6   begin
7       for i := 1 to m do   // where m is the number of subgraphs
8           for j := 1 to n do   // where n is the number of nodes in the subgraph
9                                Gi
10          calculate the information content IC(cj^i);  // where c^i ∈ Gi
11          calculate the depth D(cj^i);
12          calculate the local network density E(cj^i);
13          calculate the semantic distance dist(q, cj^i);
14          calculate the term similarity score sim(q, cj^i);
15          end-for
16      end-for
17  end
```

Fig. 2. The semantic similarity measure algorithm.

erature references. For example, gene product "rpl23-A" (*Chloroplast 50S ribosomal protein L23*, GR:P12097), an *Oryza sativa* species from Gramene (http://www.gramene.org) database, is shared among GO terms like "plastid" (GO:0009536), a cellular component that is supported by an evidence code of Inferred from Curator (IC) and a literature reference PMID:12520024; "RNA binding" (GO:0003723), a molecular function, is supported by an evidence code of inferred from Reviewed Computational Analysis (RCA) and literature reference GR.REF:8030; and "translation" (GO:0006412), a biological process, is supported by an evidence code of RCA and literature reference GR.REF:8030. These links are used to calculate the term similarity score between these three GO terms even though they are not directly connected by the "is-a" or "part-of" relationships, are from different categories, and do not have similar keywords. The information content of the GO term $IC(c)$ is represented as follows:

$$IC(c) = -\log(P(c)), \tag{3}$$

where $P(c)$ is the probability of occurrence of a GO term c in the association. The probability is measured using maximum likelihood estimation as given below:

$$P(c) = \frac{freq(c)}{N}, \tag{4}$$

where N is the total number of occurrences in the association and $freq(c)$ is the number of times that the GO term c and all its descendants occur in the association. The frequency of the GO term c is defined as follows:

$$freq(c) = \sum_{c \in descendants(c_i)} occur(c_i), \tag{5}$$

where $descendants(c)$ is a function that returns a set of GO terms that are the descendants of the GO term c. Note that if a GO term c_1 is an ancestor of a GO term c_2, then $freq(c_1) \geqslant freq(c_2)$ since the GO term c_1 subsumes the GO term c_2 and all its descendants. Therefore, $P(c)$ is larger when the GO term c is nearer to the root term c_0, and $IC(c_1) \leqslant IC(c_2)$.

3.1.2. Conceptual distance approach

The conceptual distance of a GO term is calculated based on the depth and the local network density factors. The depth is referred to as the distance of the GO term in the hierarchy of the GO graph. The local network density is related to the number of children that span out from

the GO term. The depth of the GO term $D(c)$ is given as follows:

$$D(c) = \left(\frac{d(c)+1}{d(c)}\right)^{\alpha}, \tag{6}$$

where $d(c)$ is the level of the GO term c in the GO graph. The depth of the root term c_0 is 1, and it increases as the altitude of the GO term decreases in the hierarchy. The parameter α controls the degree of how much the depth factor contributes to (6), and $\alpha \geqslant 0$.

The local network density of the GO term $E(c)$ is given by the following equation:

$$E(c) = \left((1-\beta \times \frac{\bar{E}}{e(c)}\right) + \beta, \tag{7}$$

where $e(c)$ is the number of edges that begin from the GO term c and $\bar{E}$ is the number of edges divided by the number of GO terms in the GO graph. The parameter β controls the degree of how much the local network density factor contributes to (7), and $0 \leqslant \beta \leqslant 1$. The effect of multiple inheritances is not considered in (7) since they have been considered during calculation of the information content as mentioned in (5). Furthermore, the term similarity score between GO terms c_m and c_n is calculated according to the shortest path that links both of the GO terms via their nearest shared ancestor as formulated in (8) and (9).

Note that the parameters α and β become less important when α approaches 0 and β approaches 1, since $D(c)$ and $E(c)$ will reach 1, respectively. Furthermore, (6) and (7) are equal when $\alpha = 0$ and $\beta = 1$.

3.1.3. The hybrid approach

The hybrid approach is derived from the notion of the conceptual distance, and by incorporating the information content as a decision factor. Given a sequence of GO terms $c_1, \ldots, c_n$ representing the path from GO term c_1 to c_n with length n. The hybrid approach computes the semantic distance between GO terms c_1 and c_n by the following formula:

$$dist(c_1, c_n) = \sum_{i=0}^{n-1} D(c_i) \times E(c_i) \times (IC(c_{i+1}) - IC(c_i)), \tag{8}$$

where $dist(c_1, c_n)$ is the summation of edge weights along the shortest path that links c_1 with c_n. Thence, the semantic distance between GO terms c_m and c_n is quantified as given below:

$$dist(c_m, c_n) = dist(c_1, v_m) + dist(c_1, v_n), \tag{9}$$

where GO term c_1 is the nearest shared ancestor of GO terms c_m and c_n. As the semantic distance is founded on the difference between the information content, the normalization of the semantic distance is given by:

$$dist_{norm}(c_m, c_n) = \min\left\{1, \frac{dist(c_m, c_n)}{\max\{IC(c)\}}\right\}. \tag{10}$$

Therefore, the term similarity score between GO terms c_m and c_n is measured by converting the semantic distance as follows:

$$sim(c_m, c_n) = 1 - dist_{norm}(c_m, c_n). \tag{11}$$

Note that $0 \leqslant sim(c_m, c_n) \leqslant 1$ because $0 \leqslant dist_{norm}(c_m, c_n) \leqslant 1$.

3.2. Genetic similarity algorithm

An overview of the genetic similarity algorithm is shown in Fig. 3. The genetic similarity algorithm takes the GO graph and a query GO term as an input. The best chromosome representing a set of GO terms that have higher term similarity score to the query GO term is returned by the genetic similarity algorithm. The genetic similarity algorithm uses the semantic similarity measure algorithm to calculate the term similarity score which is the semantic similarity measure between each GO term and the query GO term.

3.2.1. Preprocessing

The first step of the genetic similarity algorithm is the calculation of the term similarity score between each node in the subgraphs of the GO graph and the query GO term. The GO graph is partitioned into several subgraphs in order to make calculation of the term similarity score and generation of the initial population easier and faster. The preprocessing step is done by the semantic similarity measure algorithm to improve the quality of the chromosome. This is done by setting the positions of nodes in the chromosome before the initialization step. Thus, the first chromosome created contains the nodes with the highest term similarity score in each subgraph. The second chromosome contains the second best and so on, as shown in example in Fig. 4. Note that the GO term accession number is mapped to the node number according to the identification in the "term" table.

3.2.2. Chromosome representation

Based on the results returned by the semantic similarity measure algorithm, the initial population is generated according to the following representations: population size is the size of the subgraph with the highest node compared to other subgraphs; chromosome length is the number of nodes in the GO graph; loci represent the node number; a gene specifies whether a node in the pool of nodes is represented by a chromosome or not; and an allele is formed by two binary elements either 0 or 1, where 1 shows presence (retrieved) and 0 shows absence (not retrieved) of a node in a chromosome.

A chromosome is created by taking a node from each subgraph beginning with the ones with higher term similarity score, as shown in example in Fig. 5. If the cardinality of a subgraph is smaller than the number of chromosomes to be produced, then that subgraph will not be present in each chromosome. An example of mapping of a GO graph

```
1   Genetic-Similarity-Algorithm (G, q);
2   Input: G (a GO graph) and q (a query GO term)
3   Output: x_best (the best chromosome representing a set of GO terms that have
4   higher term similarity score to the query GO term)
5   begin
6      preprocessing by semantic similarity measure algorithm;
7      t := 0;
8      initialize Pop(t); // note that Pop(t) = {x_1^t, ..., x_ps^t} where Pop(t) and x^t
9                            are the population and chromosome for generation t
10                           respectively and ps is the size of population
11     evaluate Pop(t);
12     while not termination-condition do
13        t := t + 1;
14        select Pop(t) from Pop(t − 1);
15        alter Pop(t) by crossover and mutation operators;
16        evaluate Pop(t);
17     end-while
18  end
```

Fig. 3. The genetic similarity algorithm.

into a chromosome is shown in Fig. 6. This representation is crucial to ensure that the large GO graph can be presented with a simple and straightforward representation; the processing time taken to converge can be shortened since the chromosome is represented using 1D binary string; and the evolution of the genetic similarity algorithm is started with an initial population such that $t_1(x_i) \geqslant t_1(x_j)$, where $t_1(x)$ is the sum of the term similarity score of the nodes in a chromosome x, $\forall i,j \in \{1,2,\ldots,ps\}$, ps is the size of population, and $i < j$.

3.2.3. Crossover and mutation operators

In order to keep the genetic similarity algorithm as generic as possible, it uses normal crossover and mutation operators. These operators are chosen since they are formed effectively with a simple 1D binary string representation and with a fitness function that uses the semantic similarity measure. At each generation, the genetic similarity algorithm implements the fitness function as criteria to evaluate the goodness of each chromosome of the current population to create a new set of artificial creatures (a new population). Thence, the fitness value of the best chromosome in each generation can be maximized, as shown in example in Fig. 7.

The above objective is attained by the crossover and mutation operators that try to improve the total fitness value of the current population by fixing the old ones. Through the crossover operator, the chromosomes reproduced in the new mating pool are matched randomly and afterward each couple of chromosomes, say x_a and x_b, undergoes a cross change. Then, the mutation operator plays a secondary role to forbid an irrecoverable loss of potentially useful information which occasionally crossover can cause. This operator conducts a random alteration of the allelic value of a chromosome.

3.2.4. Fitness function

The fitness function used focuses on maximizing the preferences for term similarity score. The decision is inspired by the demand of searching for a set of GO terms with higher term similarity score that perfectly match the query GO term. The fitness function $f(x)$ for chromosome x is shown below:

$$f(x) = \chi \times t_1(x) + \delta \times t_2(x), \tag{12}$$

Step 1: Given a GO graph with 4 subgraphs and 20 nodes.

Given a query GO term "4"

Subgraph 1	Subgraph 2	Subgraph 3	Subgraph 4
1	4	2	5
3	6	7	8
9	10	12	15
13	11	16	19
20	14	18	
	17		

Step 2: Calculate the term similarity score between each node in the subgraphs and the query GO term "4".

Subgraph 1	Subgraph 2	Subgraph 3	Subgraph 4
1 (0.29)	4 (1.00)	2 (0.38)	5 (0.41)
3 (0.13)	6 (0.23)	7 (0.76)	8 (0.27)
9 (0.33)	10 (0.16)	12 (0.80)	15 (0.45)
13 (0.66)	11 (0.73)	16 (0.19)	19 (0.20)
20 (0.05)	14 (0.28)	18 (0.42)	
	17 (0.51)		

Step 3: Sort the nodes in each subgraph according to their term similarity score.

	Subgraph 1	Subgraph 2	Subgraph 3	Subgraph 4
Rank 1	13 (0.66)	4 (1.00)	12 (0.80)	15 (0.45)
Rank 2	9 (0.33)	11 (0.73)	7 (0.76)	5 (0.41)
Rank 3	1 (0.29)	17 (0.51)	18 (0.42)	8 (0.27)
Rank 4	3 (0.13)	14 (0.28)	2 (0.38)	19 (0.20)
Rank 5	20 (0.05)	6 (0.23)	16 (0.19)	
Rank 6		10 (0.16)		

Fig. 4. An example of preprocessing for a GO graph with 4 subgraphs and 20 nodes in which "4" is the query GO term.

where χ and δ are control parameters so that the contributions given by factors $t_1(x)$ and $t_2(x)$ are harmonious. The value of the fitness function is stated as a positive value that is higher for the best chromosome.

The fitness function comprises two factors. The first factor is the sum of the term similarity score of the nodes in chromosome x, and is given as follows:

$$t_1(x) = \sum_{u_i \in x} score(u_i), \tag{13}$$

where $score(u_i)$ is the term similarity score between the query GO term and nodes that are present in chromosome x. This factor considers the positive effect of having as many nodes with high term similarity score as possibly present in a chromosome. Nonetheless, a chromosome with many nodes with low score could create a fitness value higher than another one with a few good nodes. To avoid this consequence, the dimension index $t_2(x)$ is introduced as follows:

$$t_2(x) = \frac{k}{abs(cnt(x) - ID) + 1}, \tag{14}$$

where k is the number of nodes in the GO graph, $cnt(x)$ is the number of nodes present in chromosome x, and ideal dimension (ID) is the number of matched GO terms that are preferred to be returned to the user. Note that $0 < t_2(x) \leqslant k$ since if the number of nodes present in chromosome x is exactly equal to the ideal dimension, then maximum k is reached. Otherwise, it is rapidly lessened when the number of nodes present in chromosome x is smaller or greater than the ideal dimension.

3.2.5. Parallelization process

The major computational challenge of searching a group of semantically similar GO terms is the size of the search space of the GO graph because the GO graph has almost 23 thousand nodes and almost 2.0 million paths. Moreover, to obtain a good solution, it requires a multitude of chromosomes, many generations of population, and it undergoes several iterations of the genetic operation by crossover and mutation operators. To overcome these matters, the genetic similarity algorithm is parallelized by exploiting the advantages of the island (coarse-grained) model [36–38] as shown in Fig. 8. It is implemented on a low-cost PC cluster using message passing interface libraries. The core process of the parallelization is to divide the population into equal size subpopulations. Hereafter, each subpopulation is assigned to a processor where it evolves independently. During the process, a group of best chromosomes called emigrants are transferred to replace a group of worst chromosomes among the subpopulations. This migration process is performed periodically at certain

Subgraphs and initial population mapping.

	Subgraph 1	Subgraph 2	Subgraph 3	Subgraph 4
Rank 1	13 (0.66)	4 (1.00)	12 (0.80)	15 (0.45)
Rank 2	9 (0.33)	11 (0.73)	7 (0.76)	5 (0.41)
Rank 3	1 (0.29)	17 (0.51)	18 (0.42)	8 (0.27)
Rank 4	3 (0.13)	14 (0.28)	2 (0.38)	19 (0.20)
Rank 5	20 (0.05)	6 (0.23)	16 (0.19)	
Rank 6		10 (0.16)		

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Chromosome 1	0	0	0	1	0	0	0	0	0	0	0	1	1	0	1	0	0	0	0	0
Chromosome 2	0	0	0	0	1	0	1	0	1	0	1	0	0	0	0	0	0	0	0	0
Chromosome 3	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	1	0	0
Chromosome 4	0	1	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0
Chromosome 5	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	1
Chromosome 6	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0

Fig. 5. An example of generating initial population. Note that subgraphs "1" and "3" are not present in chromosome "6" and subgraph "4" is not present in chromosome "5" and "6" since their cardinality is smaller than the size of population.

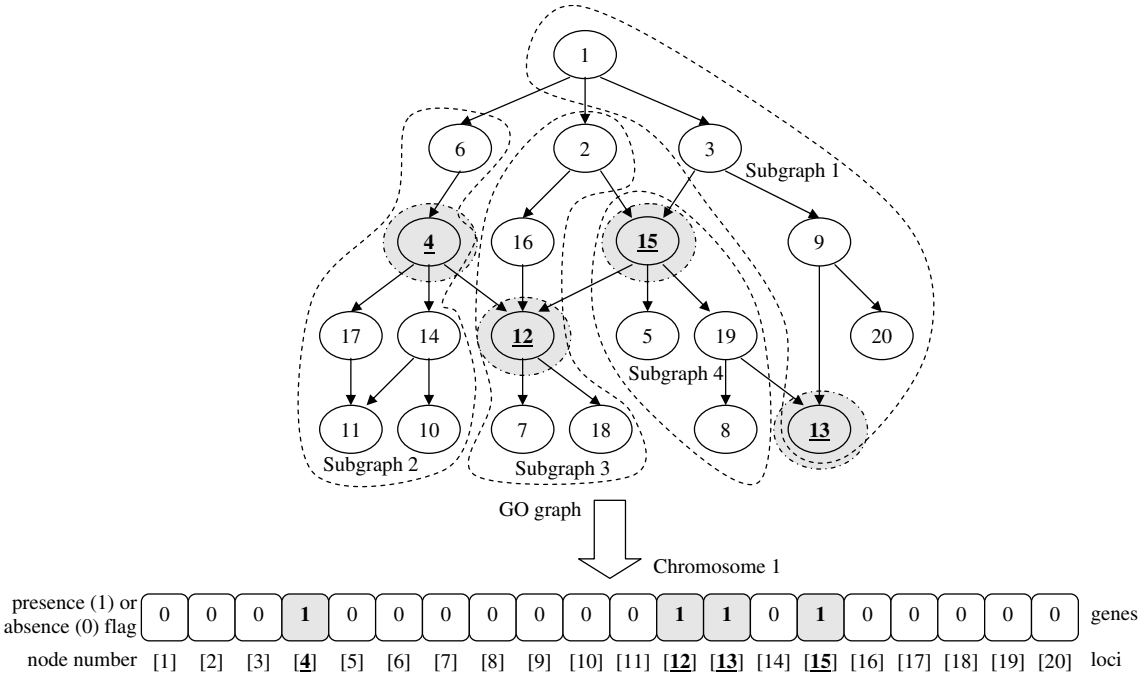

Fig. 6. An example of mapping of a GO graph into a chromosome. The mapping of nodes "13", "4", "12", and "15" with the highest term similarity score from subgraphs "1", "2", "3", and "4", respectively into chromosome "1".

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	$f(x)$ value
Generation 0	0	0	0	1	0	0	0	0	0	0	0	1	1	0	1	0	0	0	0	0	145.77
Generation 10	0	0	0	0	1	0	1	0	1	0	1	0	0	1	0	0	0	0	1	0	202.71
Generation 20	0	0	0	0	1	0	1	0	1	0	1	0	0	1	0	0	0	0	1	0	202.71
Generation 30	0	0	0	0	1	0	1	0	1	0	1	0	0	1	0	0	0	0	1	0	202.71
Generation 40	0	0	0	0	1	1	1	1	1	0	1	0	0	1	0	0	0	0	1	0	336.54
Generation 50	0	0	0	0	1	1	1	1	1	0	1	0	0	1	0	0	0	0	1	0	336.54
Generation 60	0	0	0	1	1	1	1	1	1	0	1	0	0	1	0	0	0	0	0	0	337.34
Generation 70	0	0	0	1	1	1	1	1	1	0	1	0	0	1	0	0	0	0	0	0	337.34
Generation 80	0	0	0	1	1	1	1	1	1	0	1	0	0	1	0	0	0	0	0	0	337.34
Generation 90	0	1	0	1	1	1	1	1	0	0	1	0	1	1	0	0	0	0	0	0	504.72
Generation 100	0	1	0	1	1	1	1	1	1	0	1	0	1	0	0	0	0	1	0	0	1,005.19

Fig. 7. An example of the best chromosome produced by mutation and crossover operators. Note that the evolution stopped after a convergence occurred at 100 generations, the fitness value of the best chromosome is 1005.19, and the best chromosome returns {"2","4", "5", "6", "7", "8", "9", "11", "13", "18"} as a set of GO terms that semantically similar to the query GO term "4".

cycles of generations called isolation time. The rationale for implementing the island model is to reduce the execution time by decreasing the communication overhead involved in the exchange of chromosomes between processors, and to improve the quality of the solutions reached by increasing population sizes without increasing the time complexity.

4. Materials and implementations

4.1. Environment preparation and data sets

The testing is executed using a low-cost PC cluster that consists of 25 Pentium IV 2.8 GHz processors with 512 MB RAM and 100 Mbps NIC. The operating system used is Fedora Core 5. The low-cost PC cluster is implemented using MPICH2 libraries [39] developed by the Argonne National Laboratory. This setup is the minimum requirement for offline usage if the user wants to install and use the *basic* and the *extended* UTMGO locally. However, for online usage, these tools can be accessed remotely via the internet like other online bioinformatics tools. But currently these tools are not ready for online usage and will be opened for public soon.

In this study, the GO data released in January 2007, as shown in Table 1, is explored in the experiments. The full GO graph that consists of 22,954 GO terms (1977 cellular components, 12,903 biological processes, and 8074 molecular functions) is input to the genetic similarity algorithm and it becomes the chromosome length. The parameters

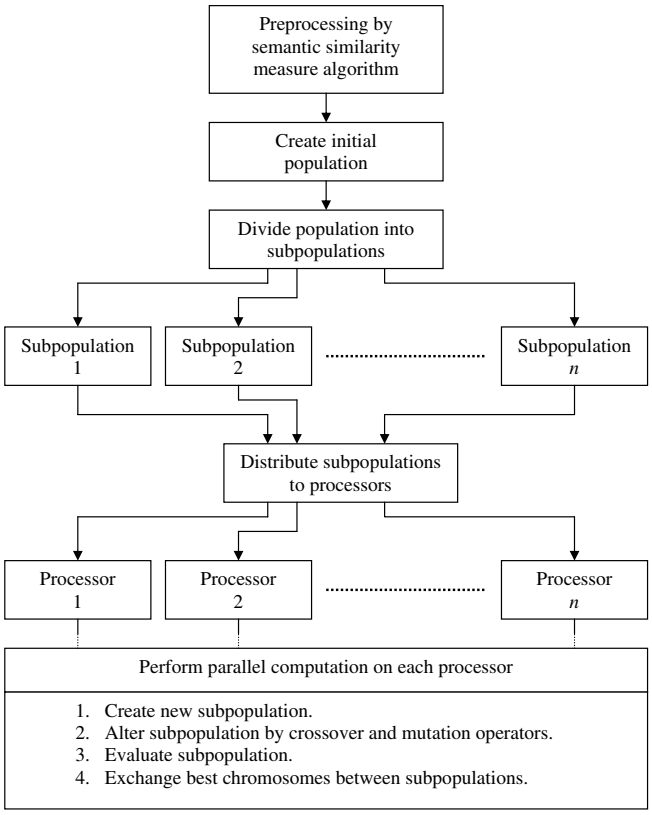

Fig. 8. The parallelization flow of the genetic similarity algorithm.

Table 1
Size of the GO data

Item	No. of records
GO terms	22,954
Definitions of GO terms	22,086
Synonyms for GO terms	20,797
Relationships between GO terms	35,006
All paths in GO graph	1,970,267
External database identifier entities	5,833,963
Links from GO terms to other databases	92,670
Gene products	2,498,910
Synonyms for gene products	330,752
Link between gene product and GO term	10,380,867
Gene product counts per GO term	550,392
Evidence type and reference for an association between gene product and GO term	11,866,795
External database links for an association between gene product and GO term	11,436,198
Protein sequences	2,310,180
Link between gene product and protein sequence	2,315,391
External database links for a protein sequence	21,761,312
Species	268,435

set for the genetic similarity algorithm are depicted in Table 2. A total of 250 GO terms in which 20 GO terms from cellular components, 140 GO terms from biological processes, and 90 GO terms from molecular functions were selected randomly as the query GO terms to evaluate the performance of the *basic* UTMGO and its genetic similarity algorithm. In case of the *extended* UTMGO, a total of 200 protein sequences from the GO annotated databases

Table 2
Parameters of the genetic similarity algorithm

Parameter	Value
Size of population	500
Number of generations	1000
Crossover probability	0.6
Mutation probability	0.05
Length of chromosome	22,954
Replacement percentage	0.5
Type of crossover	Two-point crossover
Type of mutation	Swap mutation
Type of genetic algorithm	Steady-state genetic algorithm
Scaling	Sigma truncation scaling
Fitness function	Maximizing preferences
Isolation time	10 generations
Number of subpopulations	25
Number of emigrants	1
Type of replacement	Bad by best
Type of migration	Stepping stone
Parameter α for depth factor	0.5
Parameter β for local network density factor	0.3
Parameter χ for fitness function	1
Parameter δ for fitness function	0.05
Ideal dimension for dimension index	20

were used as input. These protein sequences were selected randomly with 50 protein sequences from Gramene, a database of *Oryza sativa*; 50 protein sequences from Ensembl (http://www.ensembl.org), a database of *Homo sapiens*; 50 protein sequences from *Saccharomyces* Genome Database (SGD; http://www.yeastgenome.org), a database of *Saccharomyces cerevisiae*; and 50 protein sequences from TAIR (http://www.arabidopsis.org), a database of *Arabidopsis thaliana*.

The effectiveness of the *basic* and the *extended* UTMGO is validated using standard information retrieval measures: recall and precision. Recall is the ratio of the number of relevant GO terms retrieved to the total number of relevant GO terms in the GO database. Precision is the number of relevant GO terms retrieved to the total of irrelevant and relevant GO terms retrieved. These are formulated as:

$$\text{Recall} = \frac{a}{(a+b)} \times 100 \text{ and,} \tag{15}$$

$$\text{Precision} = \frac{a}{(a+c)} \times 100, \tag{16}$$

where a is the number of relevant GO terms retrieved (i.e., the system and the expert agree with the matches), b is the number of relevant GO terms not retrieved (i.e., the system disagrees with the matches but the expert agrees), c is the number of irrelevant GO terms retrieved (i.e., the expert disagrees with the matches but the system agrees), and d is the number of irrelevant GO terms not retrieved (i.e., the system and the expert disagrees with the matches). Here, the expert refers to a biologist who has knowledge of the GO and protein sequence annotations and the system refers to the *basic* and the *extended* UTMGO.

4.2. The basic and the extended UTMGO

In order to show the practicality of this study, we present the *basic* UTMGO, a tool that uses genetic similarity algorithm to find a group of semantically similar GO terms. The structure of the *basic* UTMGO is extended to demonstrate how it can be applied to annotate anonymous protein sequences. A screenshot of the *basic* UTMGO is shown in Fig. 9 wherein "DNA binding" (GO:0003677) is used as an example of the query GO term. A brief explanation of the processing behind the *basic* UTMGO is as follows:

i. Public GO data in MySQL and RDF/XML formats are downloaded from the GO website.
ii. The single humongous GO RDF/XML file is split into smaller files.
iii. Corresponding gene products together with protein sequences and evidence associations with the GO terms, either based on IEA or non-IEA evidence code, from the GO MySQL database are inserted into the fragmented GO RDF/XML files.
iv. The *basic* UTMGO requires the user to enter a GO term and the number of matched GO terms to be returned N_{t1}.
v. The semantic similarity searching is performed by the genetic similarity algorithm. The results return N_{t1} GO terms with higher term similarity score to the query GO term. The information displayed to the user is the GO terms accession number, followed by a short description of the GO term, its category (either cellular component (*C*), molecular function (*F*), or biological process (*P*)), and the term similarity score.

The operation of the *extended* UTMGO is divided into two cases: with (Option 1) or without (Option 2) a GO term entered by the user as shown in Figs. 10 and 11, respectively. An example of the query anonymous protein sequence used to demonstrate the *extended* UTMGO is as follows:

MVRGKTQMKRIENPTSRQVTFSKRRNGLLKKA
FELSVLCDAEVALIVFSPRGKLYEFASASTQKTIERY
RTYTKENIGNKTVQQDIEQVKADADGLAKKLEAL
ETYKRKLLGEKLDECSIEELHSLEVKLERSLISIRGR
KTKLLEEQVAKLREKEMKLRKDNEELREKCKNQP
PLSAPLTVRAEDENPDRNINTTNDNMDVETELFIG
LPGRSRSSGGAAEDSQAMPHS

This protein sequence belongs to "MADS50" (*MADS-box transcription factor 50*, GR:Q9XJ60), an *Oryza sativa* species obtained from the Gramene database. The *extended* UTMGO, as shown in Fig. 1, consists of the following steps:

i. Get an anonymous protein sequence, the number of GO terms to be returned N_{t2}, a term similarity threshold, the number of protein sequences associated with each GO term to be returned N_s, and optionally a GO term from the user.
ii. If the GO term is null, then go to step (iii), otherwise, go to step (vi).
iii. Get the input from the user for appropriate species, matrix type either Blocks Substitution Matrix (BLOSUM) or Point Accepted Mutations (PAM), and open and extend gap penalties to restrict the search.
iv. Perform the sequence similarity search for the query anonymous protein sequence from step (i). The search is carried out for protein sequences from the fragmented GO RDF/XML files that are related to the molecular function terms. The output is a protein sequence with the highest sequence alignment score. The JAligner engine is used to perform the sequence similarity search.
v. Select a molecular function term with the highest association with the protein sequence obtained in step (iv) for the next step. If there is more than one term, the user has to make the selection.

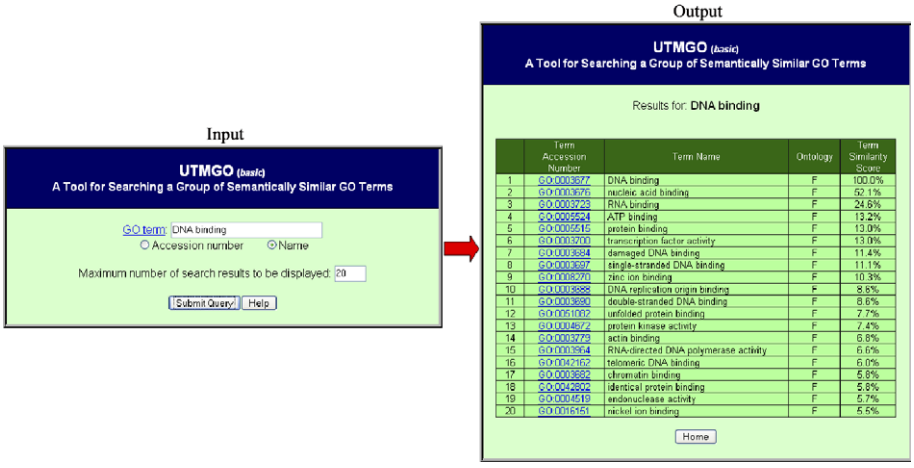

Fig. 9. A screenshot of the *basic* UTMGO.

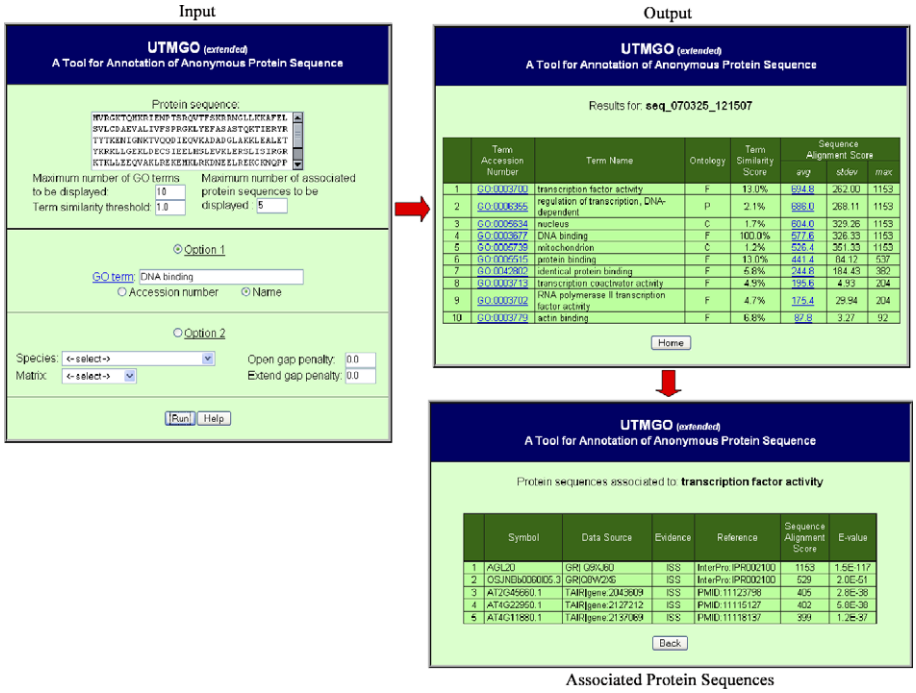

Fig. 10. A screenshot of the *extended* UTMGO with a GO term entered by the user (Option 1).

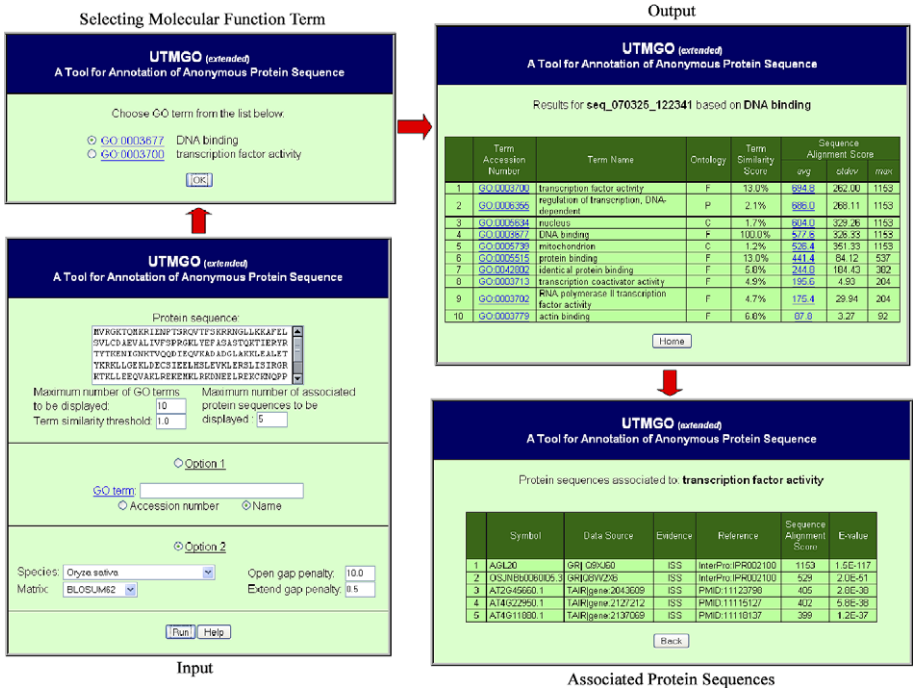

Fig. 11. A screenshot of the *extended* UTMGO without a GO term entered by the user (Option 2).

vi. Submit the GO term either from step (i) or step (v) to the *basic* UTMGO and then perform semantic similarity search.

vii. Return N_{t2} GO terms with the term similarity score higher than the term similarity threshold, as set in step (i), together with protein sequences associated with them.

viii. Calculate sequence alignment score between the query anonymous protein sequence and all protein sequences for each GO term obtained from the previous step using the JAligner engine. The information displayed to the user is the same as in the *basic* UTMGO: the GO term accession number and its short description, category, and term similarity score.

Additional information given is arithmetic mean, standard deviation, and the largest value of the sequence alignment score of N_s number of protein sequences with higher sequence alignment score that is attached to the GO term.

5. Results and discussion

Different semantic similarity measures proposed by Lin (sim_L), Leacock and Chodorow (sim_{LC}), and Resnik (sim_R) are used to assess the performance of our semantic similarity measure (sim_O) that has been built according to the Jiang and Conrath semantic similarity measure. The average results of the 250 query GO terms, as shown in Table 3, show that sim_O provides the best values of recall, precision, and maximum value of fitness function, i.e., 70.35%, 83.80%, and 1,034.02, respectively. However, the earliest number of generations to converge is obtained by sim_{LC} which converged as early as after 470 generations. Again, the best processing time (0.10 s) is obtained by sim_{LC}. Table 4 shows an example of comparison of different semantic similarity measures in which each GO term is matched with "organelle inner membrane" (GO:0019866): the term similarity score is given in percentage. GO terms such as "infected host cell surface knob" (GO:0020030), "host cell nucleus" (GO:0042025), and "membrane-bound organelle" (GO:0043227) are detected by sim_O whereas these are not detected by the other semantic similarity measures. Furthermore, the term similarity score for sim_O is higher than the other semantic similarity measures.

To examine the sensitivity of parameters α and β, different combinations of parameters α and β are analyzed. Based on the average results of the 250 query GO terms, the results from Table 5 confirm that the combination of $\alpha = 0.5$ and $\beta = 0.3$, used in this study as shown in Table 2, outperform other combinations. In the meantime, in order to justify the need for executing the genetic similarity algorithm on a low-cost PC cluster, the effect of using different numbers of processors in the low-cost PC cluster is analyzed. The effects on the following factors are studied: processing time, number of generations to converge, maximum value of fitness function, recall, and precision. The average results of the 250 query GO terms, shown in Table 6, show that a cluster of 25 processors is the ideal solution to handle the computational problem. Five factors, particularly the processing time, were highly affected if more processors were removed. Otherwise, additional processors only slightly affected those factors.

Table 3
Comparison of genetic similarity algorithm with different semantic similarity measures

Item	sim_O	sim_L	sim_{LC}	sim_R
Processing time (s)	0.13	0.16	0.10	0.11
Number of generations to converge	540	610	470	490
Maximum value of fitness function	1034.02	945.58	889.08	827.10
Recall	70.35	66.43	64.71	62.50
Precision	83.80	78.19	74.92	69.93

To prove the capability of the *basic* UTMGO that uses the genetic similarity algorithm as its intelligent engine, its output is compared with other GO browsers. The comparison is done with keyword-based GO browsers such as AmiGO, GenNav, QuickGO, and TAIR Keyword Browser, and also with semantic similarity-based GO browsers such as DynGO and FuSSiMeG. The performance is shown in Table 7 for the average results of the 250 query GO terms. Hence, the *basic* UTMGO showed better recall and precision, but the AmiGO gives the best processing time (0.11 s) which is 0.02 s faster than the *basic* UTMGO. Nevertheless, the AmiGO provides the lowest recall (54.96%) and its precision is 21.96% lower than the *basic* UTMGO. The results also show that the semantic similarity-based GO browsers outmatched the keyword-based GO browsers in terms of recall and precision. An example of a query that is based on "DNA binding" (GO:0003677) as the input GO term is shown in Table 8 (for the first 10 returned GO terms). Our semantic similarity measure is used to calculate the term similarity score: the value is given in percentage. The results from Table 8 show that all GO terms with term similarity score equal or higher than "DNA replication origin binding" (GO:0003688, 8.6%) are returned and descendingly sorted by the *basic* UTMGO. The results generated by the semantic similarity-based GO browsers are attractive because they return GO terms that do not comprise keywords associated with the query GO term. For example, "transcription factor activity" (GO:0003700), "endonuclease activity" (GO:0004519), and "protein kinase activity" (GO:0004672) are returned by the *basic* UTMGO, DynGO, and FuSSiMeG, respectively.

The comparison between the *extended* UTMGO and the other GO-based protein sequence annotation tools such as GoFigure, GOtcha, GOPET, and JAFA is shown in Table 9. The comparison is based on the average results of the 200 query protein sequences that are selected randomly from the GO annotated databases as mentioned earlier in Section 4.1. Thus, the *extended* UTMGO provides a better precision (90.32%) and the JAFA offers a better recall (88.80%) which is just 0.87% higher than the *extended* UTMGO. However, the JAFA provides the slowest processing time (518.22 s) and its precision is 3.55% lower than the *extended* UTMGO. The best processing time is 163.79 s that is taken by the *extended* UTMGO. An example query that is based on "MADS50" (*MADS-box transcription factor 50*, GR:Q9XJ60) as the input protein sequence is shown in Table 10 (for the top 10 predicted GO terms). The average and the maximum values of the sequence alignment score (avg and max) for the protein sequences associated with the predicted GO terms are used as an indicator to assess these tools, because quality of the results depends on the sequence alignment score between the query anonymous protein sequence and the protein sequences

Table 4
An example of comparison of different semantic similarity measures

GO term accession no.	GO term name	sim_O	sim_L	sim_{LC}	sim_R
GO:0005652	Nuclear lamina	5.7	4.0	3.4	2.3
GO:0005787	Signal peptidase complex	5.8	4.1	3.6	2.3
GO:0009528	Plastid inner membrane	16.1	7.7	6.5	4.3
GO:0009529	Plastid intermembrane space	1.6	1.1	0.9	0.5
GO:0009536	Plastid	9.1	5.8	2.7	0.5
GO:0016023	Cytoplasmic membrane-bound vesicle	6.5	4.3	2.1	0.5
GO:0017090	Meprin A complex	5.8	3.0	2.6	2.3
GO:0019815	B cell receptor complex	6.5	3.3	3.0	2.9
GO:0019866	Organelle inner membrane	100.0	100.0	100.0	89.0
GO:0019867	Outer membrane	7.8	7.5	6.0	4.9
GO:0020006	Parasitophorous vacuolar membrane network	4.0	2.3	1.9	1.7
GO:0020007	Apical complex	2.2	2.0	1.7	1.1
GO:0020016	Flagellar pocket	2.2	1.9	1.0	0.5
GO:0020030	Infected host cell surface knob	2.8	0.0	0.0	0.0
GO:0020031	Polar ring of apical complex	1.8	1.4	1.3	1.1
GO:0030134	ER to Golgi transport vesicle	3.9	2.8	1.4	0.5
GO:0030386	Ferredoxin:thioredoxin reductase complex	1.6	1.1	0.9	0.5
GO:0031090	Organelle membrane	12.4	10.1	8.6	3.0
GO:0031300	Intrinsic to organelle membrane	8.8	5.5	4.8	3.0
GO:0031471	Ethanolamine degradation polyhedral organelle	1.6	1.2	0.9	0.5
GO:0042025	Host cell nucleus	5.1	0.0	0.0	0.0
GO:0042601	Endospore-forming forespore	1.8	1.6	1.5	1.1
GO:0042995	Cell projection	6.4	5.0	4.2	1.1
GO:0043227	Membrane-bound organelle	12.7	0.0	0.0	0.0
GO:0043231	Intracellular membrane-bound organelle	13.8	8.2	4.0	0.5

Each GO term is matched with "organelle inner membrane" (GO:0019866), the term similarity score is in percentage.

Table 5
The effects of different combinations of parameters α and β on the values of the recall (r), precision (p), and maximum value of fitness function (f)

Parameter α for depth factor	Parameter β for local network density factor				
	$\beta = 1.0$	$\beta = 0.7$	$\beta = 0.5$	$\beta = 0.3$	$\beta = 0.0$
$\alpha = 2.0$	$r = 67.97$ $p = 81.42$ $f = 796.70$	$r = 68.19$ $p = 81.64$ $f = 818.35$	$r = 68.63$ $p = 82.08$ $f = 862.61$	$r = 68.85$ $p = 82.30$ $f = 884.79$	$r = 67.37$ $p = 80.82$ $f = 736.53$
$\alpha = 1.5$	$r = 68.88$ $p = 82.33$ $f = 887.32$	$r = 69.79$ $p = 83.24$ $f = 978.32$	$r = 69.80$ $p = 83.25$ $f = 979.31$	$r = 70.16$ $p = 83.61$ $f = 1015.50$	$r = 68.03$ $p = 81.48$ $f = 802.84$
$\alpha = 1.0$	$r = 69.14$ $p = 82.58$ $f = 912.77$	$r = 69.81$ $p = 83.26$ $f = 980.03$	$r = 69.94$ $p = 83.39$ $f = 993.91$	$r = 70.27$ $p = 83.72$ $f = 1026.49$	$r = 68.40$ $p = 81.85$ $f = 839.83$
$\alpha = 0.5$	$r = 69.35$ $p = 82.80$ $f = 934.13$	$r = 69.93$ $p = 83.38$ $f = 992.78$	$r = 70.04$ $p = 83.49$ $f = 1003.12$	$r = 70.35$ $p = 83.80$ $f = 1034.02$	$r = 68.66$ $p = 82.11$ $f = 865.45$
$\alpha = 0.0$	$r = 67.02$ $p = 80.47$ $f = 701.46$	$r = 67.65$ $p = 81.10$ $f = 764.67$	$r = 67.76$ $p = 81.21$ $f = 775.71$	$r = 68.68$ $p = 82.13$ $f = 867.52$	$r = 66.94$ $p = 80.39$ $f = 693.96$

Table 6
The effects of different number of processors used on the performance of the genetic similarity algorithm

Item	Number of processors					
	5	10	15	20	25	30
Processing time (s)	2372.37	1053.85	374.96	68.07	0.13	0.12
Number of generations to converge	690	660	610	580	540	540
Maximum value of fitness function	717.05	781.18	836.51	898.62	1034.02	1034.09
Recall	67.26	67.72	68.40	69.53	70.35	70.39
Precision	80.22	81.08	81.69	82.55	83.80	83.86

Table 7
Comparison of performance between *basic* UTMGO and other keyword-based and semantic similarity-based GO browsers

GO Browser	Recall	Precision	Processing time (s)
basic UTMGO	70.35	83.80	0.13
AmiGO	54.96	61.84	0.11
GenNav	56.78	60.92	0.16
QuickGO	57.39	60.43	0.22
TAIR Keyword Browser	56.08	61.12	0.15
DynGO	67.88	75.04	0.19
FuSSiMeG	70.26	79.41	0.23

associated with the predicted GO terms. Thus, higher is better. As depicted in Table 10, all the GO terms with the average sequence alignment score equal or higher than "RNA polymerase II transcription factor activity" (GO:0003702, avg = 175.4) are returned by the *extended* UTMGO. However, even though the average sequence alignment scores for "flower development" (GO:0009908, avg = 113.0) and "cytoplasm" (GO:0005737, avg = 93.8) are higher than "actin binding" (GO:0003779, avg = 87.8), they are out of the *extended* UTMGO radar since their term similarity scores are 0.9% and 0.6%, respectively. These term similarity scores are lower than the term similarity threshold (1.0%) set for this testing session. Moreover, as shown in Table 10, all GO terms with the highest value of the maximum of sequence alignment score (1153) are returned by the *extended* UTMGO. Note that although GO terms such as "positive regulation of transcription from RNA polymerase II promoter" (GO:0045944, max = 153), "DNA bending activity" (GO:0008301, max = 153), and "regulation of transcription from RNA polymerase II promoter" (GO:0006357, max = 151) have the maximum of sequence alignment score higher than "actin binding" (GO: 0003779, max = 92), but they are not ranked as the predicted GO terms by the *extended* UTMGO. The reason is that their average sequence alignment score is lower than the value for "actin binding" (GO: 0003779).

Determining which GO terms are relevant from over 20 thousand GO terms is not an easy task to execute, especially when what is relevant can be very subjective. A ranking function that determines the ordering of the query results, in order to determine how relevant a GO term is, is required for a basic calculation to accurately estimate the recall and precision. Furthermore, an adaptive mechanism that is capable of automatically determining the optimal values of genetic algorithm parameters such as crossover probability, mutation probability, and replacement percentage is also required. This is due to the fact that the most suitable combination of parameters for one problem or data set is not always optimal for others. In the meantime, as the size of the GO increases, additional computing resources are required to provide faster results. Understanding of the GO terms and their properties by the users is also required in order for them to use the *basic* and the *extended* UTMGO efficiently.

Table 8
An example of comparison between *basic* UTMGO and other keyword-based and semantic similarity-based GO browsers where the query GO term is "DNA binding" (GO:0003677)

Rank	*basic* UTMGO		AmiGO		GenNav		QuickGO		TAIR Keyword Browser		DynGO		FuSSiMeG	
	GO term accession no.	Term similarity score	GO term accession no.	Term similarity score	GO term accession no.	Term similarity score	GO term accession no.	Term similarity score	GO term accession no.	Term similarity score	GO term accession no.	Term similarity score	GO term accession no.	Term similarity score
1	GO:0003677	100.0	GO:0003680	5.4	GO:0003680	5.4	GO:0003677	100.0	GO:0003680	5.4	GO:0003677	100.0	GO:0003677	100.0
2	GO:0003676	52.1	GO:0050692	1.9	GO:0003681	4.2	GO:0006260	3.4	GO:0003677	100.0	GO:0005524	13.2	GO:0003676	52.1
3	GO:0003723	24.6	GO:0003677	100.0	GO:0019237	4.6	GO:0051880	2.5	GO:0003681	4.2	GO:0005515	13.0	GO:0004672	7.4
4	GO:0005524	13.2	GO:0051880	2.5	GO:0031490	3.6	GO:0003899	5.3	GO:0003684	11.4	GO:0003688	8.6	GO:0003697	11.1
5	GO:0005515	13.0	GO:0003681	4.2	GO:0003684	11.4	GO:0003887	5.0	GO:0003690	8.6	GO:0008534	3.0	GO:0008270	10.3
6	GO:0003700	13.0	GO:0019237	4.6	GO:0050692	1.9	GO:0050692	1.9	GO:0003691	3.7	GO:0003691	3.7	GO:0005515	13.0
7	GO:0003684	11.4	GO:0031490	3.6	GO:0003677	100.0	GO:0003908	3.3	GO:0003692	4.6	GO:0031490	3.6	GO:0019237	4.6
8	GO:0003697	11.1	GO:0003684	11.4	GO:0003690	8.6	GO:0003964	6.6	GO:0003695	3.1	GO:0003681	4.2	GO:0003908	3.3
9	GO:0008270	10.3	GO:0003690	8.6	GO:0003691	3.7	GO:0008534	3.0	GO:0000182	4.9	GO:0050692	1.9	GO:0042162	6.0
10	GO:0003688	8.6	GO:0003691	3.7	GO:0051880	2.5	GO:0003886	3.0	GO:0003696	5.2	GO:0004519	5.7	GO:0003682	5.8

Table 9
Comparison of performance between *extended* UTMGO and other GO-based protein sequence annotation tools

GO-based protein sequence annotation tool	Recall	Precision	Processing time (s)
extended UTMGO	87.93	90.32	163.79
GoFigure	83.15	84.09	195.48
GOtcha	83.62	84.63	302.11
GOPET	86.39	85.31	270.82
JAFA	88.80	86.77	518.22

6. Conclusions

The *basic* UTMGO is based on the genetic similarity algorithm. It is a combination of genetic and semantic similarity search, and has been presented as an alternative way of searching the GO terms. The search is done by determining a group of semantically similar GO terms that are related to the query GO term. The semantic similarity search is not based on keyword matching but is based on the degree of relationships between the GO terms. A gene product that is associated with one or more GO terms is used as a foundation to compute the amount of information the GO terms share in common that gives the degree of relationships. In the meantime, the genetic search plays the main role in finding a set of GO terms from the large GO graph. The search results have indicated that the *basic* UTMGO is able to find a group of semantically similar GO terms with higher recall and precision and reasonable processing time as compared to other existing GO browsers. The usefulness of the *basic* UTMGO has been shown by its extended version. The *extended* UTMGO has the capability of annotating anonymous protein sequences with higher precision and recall with quicker processing time. The protein sequences associated with the predicted GO terms that are returned by the *extended* UTMGO also have higher sequence alignment score to the query anonymous protein sequence. In addition, the *extended* UTMGO does not depend on BLAST and RDBMS and is fully based on the GO data. Future improvements in the *basic* and the *extended* UTMGO are to provide the user with free text typing for entering the query GO term and to develop a thesaurus for the user to check the predicted annotation. Specifically for the *basic* UTMGO, future development direction is to implement it to predict protein function and protein–protein interactions. For the *extended* UTMGO, additional enhancement includes the ability to support more than one protein sequence per query and to accept DNA sequence as an input.

Acknowledgments

This project is funded by Malaysian Ministry of Science, Technology, and Innovation (MOSTI) in part under Intensification of Research in Priority Areas (IRPA) Grants (Project No. 04-02-06-0057-EA001 and 04-02-06-10050-EAR) and in part under Short Term Research (STR) Grant (Project No. 75162).

Table 10
An example of comparison between *extended* UTMGO and other GO-based protein sequence annotation tools where the query anonymous protein sequence is "MADS50" (*MADS-box transcription factor 50*, GR:Q9XJ60), avg, average of sequence alignment score; and max, maximum of sequence alignment score

Rank	*extended* UTMGO		GoFigure		GOtcha		GOPET		JAFA	
	GO term accession no.	Sequence alignment score	GO term accession no.	Sequence alignment score	GO term accession no.	Sequence alignment score	GO term accession no.	Sequence alignment score	GO term accession no.	Sequence alignment score
1	GO:0003700	avg = 694.8 max = 1153	GO:0003700	avg = 694.8 max = 1153	GO:0003677	avg = 577.6 max = 1153	GO:0006355	avg = 686.0 max = 1153	GO:0045944	avg = 86.4 max = 153
2	GO:0006355	avg = 686.0 max = 1153	GO:0003677	avg = 577.6 max = 1153	GO:0030528	avg = 0.0 max = 0	GO:0003677	avg = 577.6 max = 1,153	GO:0006657	avg = 0.0 max = 0
3	GO:0005634	avg = 604.0 max = 1153	GO:0007275	avg = 0.0 max = 0	GO:0003700	avg = 694.8 max = 1153	GO:0003700	avg = 694.8 max = 1,153	GO:0004402	avg = 0.0 max = 0
4	GO:0003677	avg = 577.6 max = 1153	GO:0009908	avg = 113.0 max = 565	GO:0006139	avg = 0.0 max = 0	GO:0006139	avg = 0.0 max = 0	GO:0008362	avg = 0.0 max = 0
5	GO:0005739	avg = 526.4 max = 1153	GO:0006350	avg = 0.0 max = 0	GO:0006350	avg = 0.0 max = 0	GO:0006350	avg = 0.0 max = 0	GO:0007144	avg = 0.0 max = 0
6	GO:0005515	avg = 441.4 max = 537	GO:0006355	avg = 686.0 max = 1153	GO:0006355	avg = 686.0 max = 1153	GO:0045944	avg = 86.4 max = 153	GO:0007129	avg = 36.2 max = 92
7	GO:0042802	avg = 244.8 max = 382	GO:0005634	avg = 604.0 max = 1153	GO:0005622	avg = 38.6 max = 101	GO:0006357	avg = 85.2 max = 151	GO:0007020	avg = 19.0 max = 95
8	GO:0003713	avg = 195.6 max = 204	—	—	GO:0008233	avg = 29.6 max = 148	GO:0003936	avg = 0.0 max = 0	GO:0007004	avg = 0.0 max = 0
9	GO:0003702	avg = 175.4 max = 204	—	—	GO:0005215	avg = 17.0 max = 85	GO:0008301	avg = 57.4 max = 153	GO:0007015	avg = 20.2 max = 101
10	GO:0003779	avg = 87.8 max = 92	—	—	GO:0005737	avg = 93.8 max = 134	—	—	GO:0006430	avg = 16.6 max = 83

References

[1] The Gene Ontology Consortium. The Gene Ontology (GO) project in 2006. Nucleic Acids Research 2006;34:D322–6.

[2] Leung AK, Trinkle-Mulcahy L, Lam YW, Andersen JS, Mann M, Lamond AI. NOPdb: nucleolar proteome database. Nucleic Acids Research 2006;34:D218–20.

[3] Winter C, Henschel A, Kim WK, Schroeder M. SCOPPI: a structural classification of protein–protein interfaces. Nucleic Acids Research 2006;34:D310–4.

[4] Gao G, Zhong Y, Guo A, Zhu Q, Tang W, Zheng W, et al. DRTF: a database of rice transcription factors. Bioinformatics 2006;22:1286–7.

[5] Flores S, Echols N, Milburn D, Hespenheide B, Keating K, Lu J, et al. The database of macromolecular motions: new features added at the decade mark. Nucleic Acids Research 2006;34:D296–301.

[6] Camon E, Magrane M, Barrell D, Binns D, Fleischmann W, Kersey P, et al. The Gene Ontology Annotation (GOA) project: implementation of GO in SWISS-PROT, TrEMBL, and InterPro. Genome Research 2003;13:662–72.

[7] Camon E, Magrane M, Barrell D, Lee V, Dimmer E, Maslen J, et al. The Gene Ontology Annotation (GOA) database: sharing knowledge in UniProt with gene ontology. Nucleic Acids Research 2004;32:D262–6.

[8] Guo X, Liu R, Shriver CD, Hu H, Liebman MN. Assessing semantic similarity measures for the characterization of human regulatory pathways. Bioinformatics 2006;22:967–73.

[9] Tang Y, Zheng J. Linguistic modelling based on semantic similarity relation among linguistic labels. Fuzzy Sets and Systems 2006;157:1662–73.

[10] Steichen O, Bozec CD, Thieu M, Zapletal E, Jaulent MC. Computation of semantic similarity within an ontology of breast pathology to assist inter-observer consensus. Computers in Biology and Medicine 2006;36:768–88.

[11] Maki WS, Krimsky M, Munoz S. An efficient method for estimating semantic similarity based on feature overlap: reliability and validity of semantic feature ratings. Behavior Research Methods 2006;38:153–7.

[12] Leacock C, Chodorow M. Combining local context and WordNet similarity for word sense identification. In: Fellbaum C, editor. WordNet: An Electronic Lexical Database. Cambridge: MIT Press; 1998. p. 265–83.

[13] Lin D. An information-theoretic definition of similarity. In: Proceedings of the International Conference on Machine Learning; 1998. p. 296–304.

[14] Jiang JJ, Conrath DW. Semantic similarity based on corpus statistics and lexical taxonomy. In: Proceedings of the International Conference on Research in Computational Linguistics; 1998. p. 19–33.

[15] Resnik P. Using information content to evaluate semantic similarity in a taxonomy. In: Proceedings of the International Joint Conference on Artificial Intelligence; 1995. p. 448–53.

[16] Budanitsky A, Hirst G. Evaluating WordNet-based measures of lexical semantic relatedness. Computational Linguistics 2006;32:13–47.

[17] Lord PW, Stevens RD, Brass A, Goble CA. Investigating semantic similarity measures across the gene ontology: the relationship between sequence and annotation. Bioinformatics 2003;19:1275–83.

[18] Popescu M, Keller JM, Mitchell JA. Fuzzy measures on the gene ontology for gene product similarity. IEEE/ACM Transactions on Computational Biology and Bioinformatics 2006;3:263–74.

[19] Sevilla JL, Segura V, Podhorski A, Guruceaga E, Mato JM, Martínez-Cruz LA, et al. Correlation between gene expression and GO semantic similarity. IEEE/ACM Transactions on Computational Biology and Bioinformatics 2005;2:330–8.

[20] Mitra S, Hayashi Y. Bioinformatics with soft computing. IEEE Transactions on System, Man, and Cybernetics, Part C 2006;36:616–35.

[21] Kushchu I. Web-based evolutionary and adaptive information retrieval. IEEE Transactions on Evolutionary Computation 2005;9:117–25.

[22] Pal SK, Talwar V, Mitra P. Web mining in soft computing framework: relevance, state of the art and future directions. IEEE Transactions on Neural Networks 2002;13:1163–77.

[23] Chen L, Luh C, Jou C. Generating page clippings from web search results using a dynamically terminated genetic algorithm. Information Systems 2005;30:299–316.

[24] Paul TK, Iba H. Gene selection for classification of cancers using probabilistic model building genetic algorithm. Biosystems 2005;82:208–25.

[25] Rodriguez MA, Jarur MC. A genetic algorithm for searching spatial configurations. IEEE Transactions on Evolutionary Computation 2005;9:252–70.

[26] Tamine L, Chrisment C, Boughanem M. Multiple query evaluation based on an enhanced genetic algorithm. Information Possessing and Management 2003;39:215–31.

[27] Liu H, Hu ZZ, Wu CH. DynGO: a tool for browsing and mining gene ontology and its associations. BMC Bioinformatics 2005;6:201.

[28] Couto F, Silva M, Coutinho P. Semantic similarity over the gene ontology: family correlation and selecting disjunctive ancestors. In: Proceedings of the ACM Conference on Information and Knowledge Management; 2005. p. 343–4.

[29] Khan S, Situ G, Decker K, Schmidt CJ. GoFigure: automated gene ontology annotation. Bioinformatics 2003;19:2484–5.

[30] Martin DM, Berriman M, Barton GJ. GOtcha: a new method for prediction of protein function assessed by the annotation of seven genomes. BMC Bioinformatics 2004;5:178.

[31] Vinayagam A, del Val C, Schubert F, Eils R, Glatting KH, Suhai S, et al. GOPET: a tool for automated predictions of gene ontology terms. BMC Bioinformatics 2006;7:161.

[32] Friedberg I, Harder T, Godzik A. JAFA: a protein function annotation meta-server. Nucleic Acids Research 2006;34:W379–81.

[33] Groth D, Lehrach H, Hennig S. GOblet: a platform for gene ontology annotation of anonymous sequence data. Nucleic Acids Research 2004;32:W313–7.

[34] Enault F, Suhre K, Poirot O, Abergel C, Claverie JM. Phydbac (phylogenomic display of bacterial genes): an interactive resource for the annotation of bacterial genomes. Nucleic Acids Research 2003;31:3720–2.

[35] Quevillon E, Silventoinen V, Pillai S, Harte N, Mulder N, Apweiler R, et al. InterProScan: protein domains identifier. Nucleic Acids Research 2005;33:W116–20.

[36] Takashima E, Murata Y, Shibata N, Ito M. Techniques to improve exploration efficiency of parallel self-adaptive genetic algorithms by dispensing with iteration and synchronization. Systems and Computers in Japan 2006;37:25–33.

[37] Rahul CD, Dutta A. Optimization of FRP composites against impact induced failure using island model parallel genetic algorithm. Composites Science and Technology 2005;65:2003–13.

[38] Katayama K, Hirabayashi H, Narihisa H. Analysis of crossovers and selections in a coarse-grained parallel genetic algorithm. Mathematical and Computer Modelling 2003;38:1275–82.

[39] Gropp W, Lusk E, Ashton D, Buntinas D, Butler R, Chan A, Ross R, Thakur R, Toonen B. MPICH2 user's guide version 1.0.4; 2006. <http://www-unix.mcs.anl.gov/mpi/mpich2/downloads/mpich2-doc-user.pdf/>.