An Improved Ant Colony Optimization Algorithm for Clustering Proteins in Protein Interaction Network

Jamaludin Sallim[1], Rosni Abdullah[2], Ahamad Tajudin Khader[3]
[1,2,3]School of Computer Sciences,
Universiti Sains Malaysia
11800 Penang, Malaysia.
{jamals,rosni,tajudin}cs.usm.my

Abstract--Recently, we have introduced the first version of ACOPIN, an Ant Colony Optimization (ACO) algorithm for clustering proteins in the protein interaction network (PIN). Based on the experimental results, we have discovered serious pitfalls; very difficult to define the boundaries among clusters and unable to identify the overlapping clusters. These pitfalls contribute to low accuracy of detected cluster especially for the larger size. In this paper, we improve ACOPIN to overcome these pitfalls. The improvement includes the construction tour of ant based on topological weight and functional similarity weight. Artificial ants themselves perform the 'exploration' and 'exploitation' process in the PIN with the assistance from pheromone trails. We simulate the improved version onto one Yeast datasets and the result shows that the detected clusters has very high accuracy in terms of functional homogeneity of proteins.

INTRODUCTION

Protein is one of important components in a cell. In order to make a cell functioning, protein needs to interact with other components such as DNA, RNA or other proteins. Interaction among proteins or protein-protein interaction (PPI) is detected via high-throughput methods such Yeast2-Hybrid, Mass Spectrometry and Protein Micro-Array.

The network of these PPIs, called protein interaction network (PIN) are currently investigated in terms of topology, motifs, correlation structure and modular properties that are related to function [Park C.S Functional Module]. To perform this investigation, the most popular approach is detecting the clusters in PIN, which biologically refers to 'functional modules'. Various clustering algorithms have been proposed, however due to the presence of noisy and intricate data, these algorithms have not been successful to achieve high performance of clustering process [1].

Recently, we have introduced ACOPIN [2], an Ant Colony Optimization (ACO) for clustering proteins in the protein interaction network (PIN). Based on the experimental results, we discover serious pitfalls where it was very difficult to define the cluster boundaries thus it produced low accuracy of detected clusters. In this paper, we improve the ACOPIN to overcome these pitfalls. The improvement includes the construction tour by artificial ants based on topological weight and functional similarity weight. Artificial ants themselves perform the 'exploration' and 'exploitation' with the assistance of pheromone trails update. We simulate the improved version onto one Yeast datasets and the result is very convincing and shows the highly effectiveness.

The rest of the paper is organized as follows; Section 2 presents the background and the related work. Section 3 describes ACOPIN-v1 and the pitfalls. Section 4 describes the improvement tasks and Section 5 reports part of experiment results. Finally, the conclusion of this paper is drawn in section 6.

BACKGROUND AND RELATED WORKS

Ant Colony Optimization (hereinafter known as ACO) is the one of the recent heuristic algorithm developed to solve combinatorial optimization problems such as Traveling Salesman Problem, Vehicle Routing Problem, Quadratic Assignment Problem, Job Shop Scheduling Problem, etc [3]. The original version of ACO, Ant System, was developed by Marco Dorigo and his colleagues in early 1990 [4]. ACO mimics the behavior of ants foraging that move from their nest to the food sources and return to nest in a shortest path. This mimic gives various inspirations to the researchers for designing the methods and was proven to solve the problem optimally.

```
procedure ACO
  initialiaze_parameter
  while termination_condition not met
     schedule_activities
        contructAntsolutions()
        update_pheromone()
        daemon_actions()  //optional
     end schedule_activities
   end while
end procedure
```

Fig. 1. High level view of ACO algorithm.

Since 2003, the problem of clustering proteins in PIN has been vigorously investigated. The clustering approaches such as graph-based clustering, hierarchical clustering and distance-based clustering are the common approach that have been used. Based on the capability of ACO handling routing, scheduling and optimization problem, we have tried to adapt ACO to this problem [2,5,6]. First, we study the ACO

implementation in TSP problem. From there, we gather, why don't artificial ants explore and exploit in PIN. To the best our knowledge, there is no existing ACO algorithm for clustering proteins in PIN.

ACOPIN-v1: First Version and Pitfalls

Clustering process of the first version (hereinafter known as *ACOPIN-v1*) is solely based on topological properties of PIN. The main idea of ACOPIN-v1 was to find an optimal path in a given PIN. In this solution, we described the protein-protein interactions as a connectivity graph or PIN ($G = V, E$) where nodes (or vertices, V) correspond to proteins and edges, E correspond to the interactions. This PIN is represented by the interaction matrix a_{ij}, and we calculate the distances between nodes by transforming the interaction matrix a_{ij} to distance matrix d_{ij} using Bond Energy Algorithm (BEA)[5]. We apply a TSP-problem approach where artificial ants just explore the PIN without exploiting the information of the nodes (proteins). The information on edge (distances) is the only input used. In this case, we apply the *ASDecision Rules* when at junction to choose. The GO Terms is used to validate the initial predicted clusters and based on the probability only.

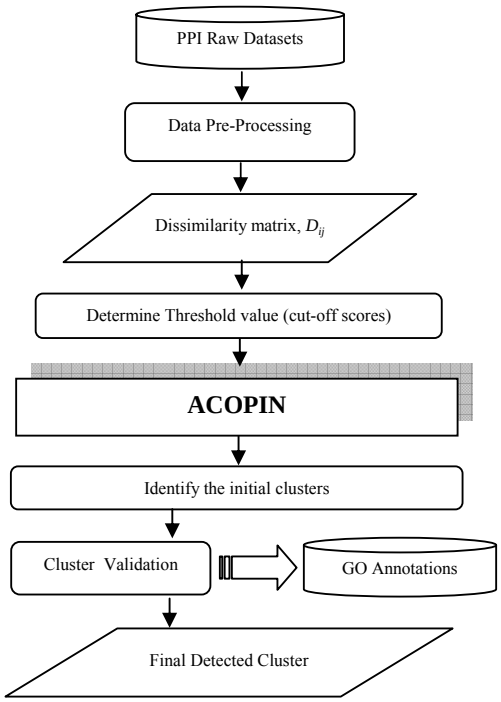

Fig. 2. The process flow of ACOPIN-v1.

The cut-off value is used to define the boundaries of protein clusters, however, if too high, the clusters contain too many proteins some of which are not similar at all; on the other hand, when the cut-off is too low, the clusters are too small. More information on ACOPIN-v1 can be found in [2]). The flow chart and the high level view algorithm for ACOPIN-v1 is shown in Figure 2 and Figure 3 accordingly.

```
procedure construct_solution
  input = PIN instance{vertex i,j, d_ij}
  output = {paths Pt, cluster C}
   for all ants;
     place_ant_randomly
     for each ant
       perform tour()
end procedure

procedure update_pheromone
   select_the_optimal_tour()
   identify_cluster_by_using_cut-off()
end procedure
```

Fig. 3. ACOPIN-v1 algorithm.

We interpret the optimal path as a clustering of PIN. We expect that proteins with similar functions are likely to be close together in the path. The reasoning for this is that proteins with similar functions are likely to interact with similar partners and so will have a short distance between them in the transformed matrix, this means they are likely to be close on the optimal path. However, the difficulty with this clustering method is that to the boundaries of the clusters are ill-defined (or in fact not defined at all) so one aspect of our work will need to be finding the optimum criteria for defining cluster boundaries.

Another serious pitfall is that ACOPIN-v1 was not capable of detecting the overlapping clusters, where a protein can be included into two or more clusters. Table 1 shows the initial results of ACOPIN-v1.

Table 1: Experimental Results For ACOPIN-v1

Datasets	# of proteins	# of interactions	# of clusters detected
S. cerevisae	2640	6600	25
H. Pylori	710	1420	14
M. Musculus	329	286	12

Table 2 shows the comparison of ACOPIN-v1 with state-of-the-art clustering algorithms, Restricted Neighborhoods Clustering (RNSC) [7] and Markov Clustering (MCL) [8]. Based on the results, ACOPIN-v1 is far away from both methods. Even though we managed to minimize the discard rate, the cluster size is big compared to the MCL and RNSC.

Table 2: Comparative Results with State-of-the-Art Algorithms on S. Cerevisae Datasets (2640 proteins and 6600 interactions)

Datasets	# of cluster detected	Average size of clusters	% discard
ACOPIN-v1	25	42.5	15.6
MCL	186	10.1	40.3
RNSC	151	8.9	26.6

ACOPIN-v2: Improved Version

In the improved version (hereinafter known as ACOPIN-v2), we added another pre-processing task where we utilize aonther properties of PIN, the functional annotation of proteins. The findings by

[9,10,11] stated that to improve the accuracy of clusters, we should combine the topological properties with other biological information that exist in PIN, such as molecular functions, biological processes or cellular organization of proteins. Since the definition of 'cluster' in PIN refers to the functional homogeneity, we only exploit the functional information of PIN. However, not all proteins are annotated and not all annotated proteins are functionally characterized. The unknown function of a annotated or unannotated proteins make this problem become more complex. With using the respective functional annotations, we calculate the functional similarity of the pairwise proteins.

A. Combining topological properties with functional information and functional similarity calculation.

The purpose of this combination is to obtain similarity and dissimilarity matrices. The topological of PIN described the dissimilarity of two adjacent proteins where the higher the value, the higher the dissimilarity or in other words, the smaller the value, the higher the similarity. The functional properties, is calculated from annotated proteins. The similarity measurement is based on the average similarity of two annotated proteins. The higher value contributes to the higher similarity. Several methods have been developed to compute the similarity of two genes products (including proteins) based on their functional annotations. In our work, we use the Lin similarity measure [12]. As done by [9], before calculating the similarity between two proteins i and j, the similarity between the terms belonging to the term sets T_i and T_j must be calculated first. These terms are used to annotate respective proteins. Given the ontology terms $t_k \in T_i$ and $t_l \in T_j$, the semantic similarity is defined as:

$$sim(t_k, t_l) = \frac{2 \ln p_{ms}(t_k, t_l)}{\ln p(t_k) + \ln p(t_l)} \quad (1)$$

where $p(t_k)$ is the probability of term t_k and $p_{ms}(t_k,t_l)$ is the probability of the *minimum subsumer* of t_k and t_l, which is defined as the lowest probability found among the parent terms shared by t_k and t_l. Given two proteins, i and j, with T_i and T_j containing m and n terms, respectively, the protein-protein similarity is defined as the average inter-set similarity between terms from T_i and T_j:

$$ss_{ij} = \frac{1}{m \times n} \sum_{t_k \in T_i, t_l \in T_j} sim(t_k, t_l) \quad (2)$$

where $sim(t_k, t_l)$ is calculated using (1).

Finally, we obtain the topological dissimilarity matrix and functional similarity matrix and these matrices will be the input to ACOPIN-v2 for clustering process.

B. Problem Formulation

Similar to ACOPIN-v1, PIN is represented in an arbitrary undirected $G = (V,E)$, where $V = \{v_1, v_2, v_3 \ldots, n\}$ is the set of vertices (proteins) of G, and E is the set of edges (interactions) of G. The distance d_{ij} between two vertices i and j in graph G is the topological properties calculation that describes the difference of common sharing partners. In the improved version, we add another distance that incorporated with PIN, Gene Ontology functional annotation where functional similarity, f_{ij} is the similarity measurement of functional annotation of annotated proteins. $f_{ij} = 0$ if one of the proteins or both of them are not annotated or doesn't have any similarity at all.

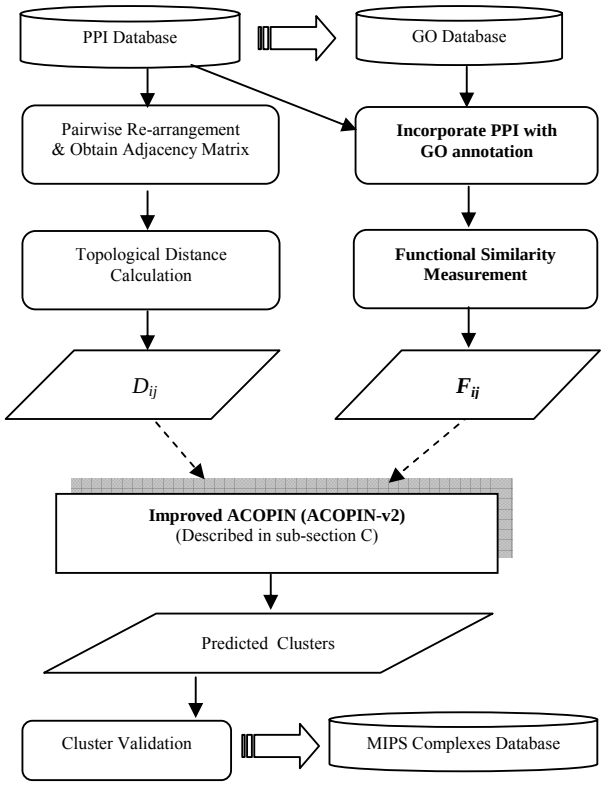

Fig. 4. The process flow of ACOPIN-v2.

C. An Improved ACOPIN (ACOPIN-v2)

The ACOPIN-v2 consists of two major phases; finding optimal path and form preliminary clusters by forward and backward ants and merge this clusters to obtain final clusters.

```
procedure ACOPIN-v2
  input = PIN instance{vertex i,j, d_ij, f_ij}
  output = {paths Pt, preliminary_cluster C^p,
            final_cluster C^f}
  initialiaze() //initialize parameters
  and pheromones
    repeat
        contruct_solutions()
        update_statistic()
```

```
      update_pheromone()
  until termination_condition met
end procedure
```

Fig. 5. High level view of ACOPIN-v2 algorithm.

The contruct solution is modified with adding 3 major tasks; optimal tour, cluster tour and merge clusters.

```
procedure construct_solution
  for all ants;
    place_ant_randomly;
    for each ant
      perform optimal_tour;
         for each path found in
         optimal_tour
              perform cluster_tour;
              form preliminary_cluster;
          end for
     end for
  end for
  merge_preliminary_cluster;
end procedure
```

Fig. 6. Construct solution algorithm.

In optimal_tour() process, forward ants are used to explore new paths in the network. They find the nearest neighbors and travel with a minimum cost. They will stop at the maximum number of nodes that they can travel. They are initiated toward randomly selected nodes hence no particular final destination node can be specified.

In cluster_tour process, a backward ant will receive the list of nodes in the respective paths from forward ants. The backward ants will travel and select the similar protein that have higher similarity. For those proteins that are not annotated or having lower similarity, they will be grouped with the nearest neighbors based on the topological measurements. Preliminary clusters also formed at this stage.

Finally, we use ants and exploit the pheromones to merge the preliminary cluster that initially formed.

D. Cluster Validation

To validate the predicted clusters by ACOPIN-v2, we apply the standard approach which is also being applied in ACOPIN-v1; by comparing the predicted cluster with the protein complexes in MIPS database [13]. During the validation, proteins that exist in the predicted cluster and also found in MIPS complexes dataset are considered valid. Otherwise, they will be discarded from the respective clusters. As a method for statistical evaluation of clusters, *p*-value was used to estimate whether a given set of proteins is accumulated by chance. *p*-value from hypergeometric distribution has been used as a criterion to assign each clster a main function. Here, we also calculated *p*-value for each identified module and assigned a function category to it when the minimum *p*-value occurred. The *p*-value was defined as the following formula:

$$P\text{-}value = 1 - \sum_{i=0}^{k-1} \frac{\binom{|F|}{i}\binom{|V|-|F|}{|M|-i}}{\binom{|V|}{|M|}} \quad (3)$$

where $|V|$ was the total number of proteins in the network, $|M|$ was the number of proteins in an identified cluster, $|F|$ was the number of proteins in a reference function, and k was the number of common proteins between the functional group and the identified cluster. Low *p*-value indicated that the module closely corresponded to the function because the network had a lower probability to produce the module by chance.

SIMULATION AND RESULTS

We simulate ACOPIN-v2 on the real PIN datasets that have gone through the pre-processing stage. The aim is to evaluate the efficacy of ACOPIN-v2 to improve the accuracy of detected clusters. The simulation tasks are divided to two categories; simulation on the small-scale data to show the operation of ACOPIN-v2 and use the similar data that experimented by previous ACOPIN to show the difference.

We use PIN datasets from [14,15] that contains 214 interactions among 70 proteins. The purpose of this simulation is to show the actual clustering operation that performed by ACOPIN-v2. Then we run ACOPIN-v2 to similar data used in ACOPIN-v1. In this simulation, first we use artificial ants (known as forward ants) to find the optimal path in PIN. We deploy multi-agent systems concept in this task where a number of ants will be placed randomly at chosen vertices and try to find the optimal path within the determined vertices. The optimal path is the shortest path that an ant will traverse and each vertex must be visited once only. Figure 5 shows the animation of this operation. Eight ants (forward ants) are placed at randomly chosen vertex.

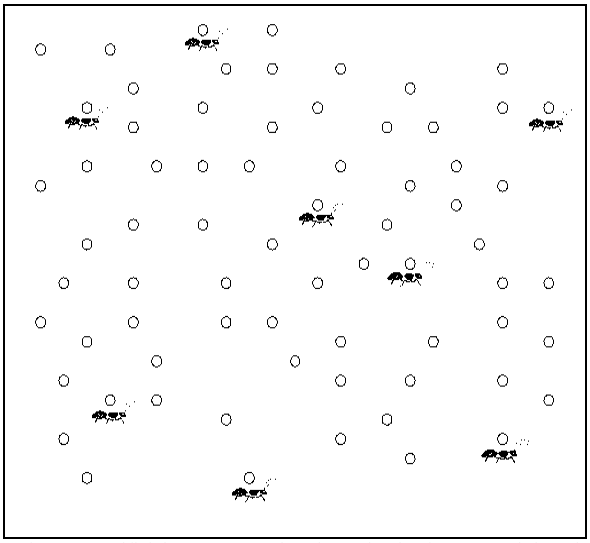

Fig. 7. Artificial ants are placed randomly in a graph-based representation of PIN data.

After ants are placed randomly in the chosen vertex, it will try to find the nearest neigbor by using choose_nearest_neighbor() function and will stop at determined number of vertex. Figure 6 shows the paths found by each ants.

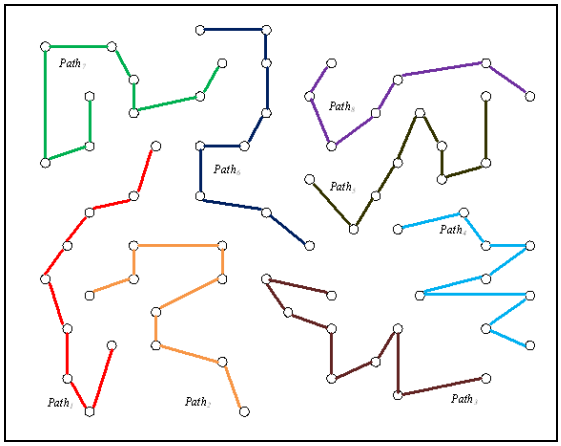

Fig. 8. The optimal paths found by *m* ants.

The number of forward or backward ants is defined based on the number of vertices in a PIN. This is to avoid the slow and computational cost in calculating and clustering data. Table 3 shows the experimental result for paths found by eight ants and the list of proteins in each path. The proteins are represented as numerical identification in pre-processing stage.

Table 3: Simulation Result. All paths with list of proteins (in sequential order)

Ant#	Path#	Proteins (represented as vertex no)
1	1	12,13,41,15,39, 33, 64,67,23
2	2	24,17,26,42,51,52,11,55,61
3	3	18,21, 2,3,16,7,40,9,28
4	4	10,1,66,34,32,54,19,20,68
5	5	35,13,14,25,43,44,5,4,6
6	6	22,27,30,49,48,37,50,51,52
7	7	46,53,29,31,70,62,63,65,69
8	8	36,47,56,57,38,58,59

Table 4 shows the detail information in each path. In this case, we print the path-1 only for the sample. We can see the proteins that grouped together with those that share common function. For unannotated proteins, we try to optimize it using function *optimize()* where they will be assigned to the nearest topological neighbors since the hypothesis stated that protein with shorter distance share higher similar partners and contribute to sharing similar functions. For instance, an unannotated protein DIP:2582N (Coactomer), has a short distance between annotated protein DIP:1318N (GTP cyclohydrolase I) and DIP:1470N (NIP1 protein). Thus they will be temporarily grouped together and will be validated in post-process.

Table 4. Preliminary clusters formed in $Path_1$.

Proteins in $Path_1$ in sequential order	A*	Functional Homogeneity Partner(s)	Topological Nearest Neighbors in $Path_1$	Common Shared Function
12 (DIP:1549N)	Y	13 (DIP:2520N) 41 (DIP:1704N) 33 (DIP:1470N)		ATP-dependent protein binding
13 (DIP:2520N)	Y	47 (DIP:2582N) 15 (DIP:1318N) 33 (DIP:1470N) 41 (DIP:1704N)		translation initiation factor activity
41 (DIP:1704N)	Y	-		
15 (DIP:1318N)	Y	23 (DIP:5181N) 64 (DIP:6580N)		serine-tRNA ligase activity
39 (DIP:2582N)	N		15 (DIP:1318N) 33 (DIP:1470N)	
33 (DIP:1470N)	Y	47 (DIP:2582N) 15 (DIP:1318N) 64 (DIP:6580N)		signal transducer activity
64 (DIP:6580N)	Y	-		
67 (DIP:6520N)	N		64 (DIP:6580N) 23 (DIP:5181N)	
23 (DIP:5181N)	Y	33 (DIP:1470N) 64 (DIP:6580N)		Rho GTPase activator activity

A* = Annotated (Y = Yes, N = No)

In post-processing tasks, first we validate the preliminary clusters using the standard assesment. If we find that the unannotated proteins are not similar at all, then the protein will be discarded.Table 5 shows the final clusters merged from the preliminary clusters.

Table 5. Final clusters* by merging preliminary clusters.

Cluster#	Size	Common Shared Function
1	7	ATP-dependent protein binding
2	4	translation initiation factor activity
3	3	
4	8	serine-tRNA ligase activity
5	4	
6	16	signal transducer activity
7	11	
8	7	
9	11	Rho GTPase activator activity
10	3	
11	5	

*for cluster size > 2

We compare this result with ACOPIN-v1 and we find that there was a huge difference in terms of number of detected clusters, especially for larger datasets. Table 6 shows the comparative result between two versions.

Table 6. Experimental results and comparison with ACOPIN-v1.

Datasets	# of proteins	# of interactions	# of clusters detected by ACOPIN-v1	# of clusters detected by ACOPIN-v2	Δ
S. Cerevisae	2640	6600	25	121	96
H. Pylori	710	1420	14	56	42
M. Musculus	329	286	12	14	2

Next step is to comparing ACOPIN-v2 with the state-of-the-art clustering algorithms. Table 7 shows the comparative results with two state-of-the-art clustering algorithms on Yeast datasets. The results show that the re is an increment in terms of number of cluster detected. The average size of clusters also increase due to the capability ACOPIN-v2 to detect the overlapping

clusters. There is a decrement on the percentage discard rate, which means ACOPIN-v2 able to cluster the proteins that are sparsely connected.

Table 7: Comparative Results with State-of-the-Art Algorithms on S. Cerevisae Datasets (2640 proteins and 6600 interactions)

Datasets	# of cluster detected	Average size of clusters	% discard
ACOPIN-v2	121	38.6	16.6
MCL	186	10.1	40.3
RNSC	151	8.9	26.6

CONCLUSIONS

In this paper, we present an improved ACO algorithm for clustering proteins in a protein interaction network. This improved version resolve the pitfalls of first version. The enhancement was achieved by adding functional information to ACOPIN. We also propose the usage of forward and backward ants for optimal path and clustering process. The experimental results demonstrated the effectiveness of the improved ACOPIN for solving this problem and showed the impact of its new features, including the construction tour by artificial ants, the usage of forward and backward ants, and management of pheromone trails.

ACKNOWLEDGEMENTS

We thank Dr Thomas Stútzle for his advice on ACO and for the permission to use and modify the original ACO source codes. We also thank both Dr Igor Jurisica and Dr Sylvian Brohée for providing us RNSC and MCL source codes. This work was supported by the Postgraduate Research Grant, Universiti Sains Malaysia (USM-RU-PGRS) under grant no. 1001/PKOMP/841007 through Research Creativity and Management Office, Universiti Sains Malaysia, Penang, Malaysia.

REFERENCES

[1] Lin, C., Cho, Y.R., Hwang, W.C., Pei, P., Zhang, A., Clustering Methods in a Protein-Protein Interaction Network. In Chapter. Knowledge Discovery in Bioinformatics (2007), 18-25.

[2] Sallim, J, Abdullah. R, Khader, A. T, ACOPIN: Ant Colony Optimization with TSP approach for Clustering Proteins in Protein Interaction Networks, Second UKSIM European Symposium on Computer Modeling and Simulation 2008, pp. 203-208.

[3] Dorigo, M., and Stutzle, T., Ant Colony Optimization, MIT Press USA 2004.

[4] Dorigo M., Maniezzo V., Colorni A., The ant systems: optimization by colony of cooperating agents, IEEE Transactions on Systems, Man, and Cybernetics-PartB, (1996).

[5] Sallim, J, Abdullah. R, Khader, A. T, Data Pre-Processing Methodology in Analyzing Protein Interaction Networks, Second Asia International Conference on Modelling & Simulation, 2008, pp. 843-848.

[6] Sallim, J, Abdullah. R, Khader, A. T, Optimal Path Finding, The International Conference on Distributed Frameworks & Applications 2008, pp. 41-44.

[7] King AD, Przulj N, Jurisica I: Protein complex prediction via cost-based clustering. *Bioinformatics* 2004, 20(17):3013-20.

[8] Enright AJ, Dongen SV, Ouzounis CA: An efficient algorithm for large-scale detection of protein families. *Nucleic Acids Res* 2002, 30(7):1575-84.

[9] Lubovac, Z., J. Gamalielsson, and B. Olsson, Combining functional and topological properties to identify core modules in protein interaction networks. Proteins, 2006. 64(4): p. 948-59.

[10] Cho, Y-R C, Hwang W. C, Zhang, A., Identification of Overlapping Functional Modules in Protein Interaction Networks: Information Flow-based Approach, Sixth IEEE International Conference on Data Mining - Workshops (ICDMW'06), 2006.

[11] Chua HN, Sung WK, Wong L, Exploiting indirect neighbours and topological weight to predict protein function from protein–protein interactions, Bioinformatics 22(13): 2006. pp1623–1630.

[12] Lin, D. An information-theoretic definition of similarity. in The 15thInternational Conference on Mashine Learning. 1998. Madison, WI.

[13] Ruepp, A., at al. The FunCat: a functional annotation scheme for systematic classification of proteins from whole genomes. *Nucleic Acid Research*, , 2004, 32(18):5539–5545

[14] Xenarios, I., et al., DIP: the database of interacting proteins. Nucleic Acids Res, 2000. 28(1): p. 289-91

[15] Database of Interacting Proteins (DIP). Available online on http://dip.doe-mbi.ucla.edu/dip/Stat.cgi.