Vol. 19 Suppl. 1 2003, pages i177–i179
DOI: 10.1093/bioinformatics/btg1022

Complexity management in visualizing protein interaction networks

Byong-Hyon Ju and Kyungsook Han*

School of Computer Science & Engineering, Inha University, Inchon 402-751, Korea

Received on January 6, 2003; accepted on February 20, 2003

ABSTRACT

Motivation: Protein-protein interaction networks often consist of thousands of nodes or more. This severely limits the utility of many graph drawing tools because they become too slow for an interactive analysis of the networks and because they produce cluttered drawings with many edge crossings.

Results: A new layout algorithm with complexity management operations in visualizing a large-scale protein interaction network was developed and implemented in a program called InterViewer3. InterViewer3 simplifies a complex network by collapsing a group of nodes with the same interacting partners into a composite node and by replacing a clique with a star-shaped subgraph. The experimental results demonstrated that InterViewer3 is one order of magnitude faster than the other drawing programs and that its complexity management is successful.

Availability: http://wilab.inha.ac.kr/protein/

Contact: khan@inha.ac.kr

Keywords: protein-protein interaction network, visualization, abstraction

INTRODUCTION

Protein-protein interactions are typically visualized as a graph where the nodes represent proteins and the edges represent the protein-protein interactions. The visualization of a graph is straightforward when dealing with a small number of nodes and edges. In practice, protein-protein interaction networks often consist of thousands of nodes, which severely limits the usefulness of many graph drawing tools for the following reasons: they produce cluttered drawings with many edge crossings or static drawings that are difficult to modify, they are too slow for an interactive analysis with large data sets, they require input data to be in a specific format rather than taking the data directly from protein-protein interaction databases. Since the ultimate value of visualizing protein interactions on a graph depends on the readability of the graph, a protein interaction network should convey its information both quickly and clearly.

Previously, we developed a force-directed layout program called InterViewer (Ju *et al.*, 2003). This paper presents a new algorithm that efficiently draws large-scale protein interaction networks in three-dimensions and improves the readability of the networks using complexity management operations. The new layout algorithm with the complexity management operations has been successfully implemented in a program called InterViewer3.

METHOD AND RESULTS

A common problem with many force-directed layout algorithms is that they become quite slow when dealing with large graphs because the layout adjustment at each step typically involves a calculation of the force between every pair of nodes. A new force-directed algorithm, which efficiently finds a layout of good quality without computing force between each pair of nodes, was developed. The algorithm (1) first finds the layout of the connected components of an entire network, (2) finds a global layout of the nodes with respect to the key nodes of the connected component, and (3) refines a local layout of the nodes within each connected component. The complexity of large-scale networks is managed by the following operations:

1. *Collapse a clique into a star-shaped subgraph.* A clique in an undirected graph $G = (V, E)$ is a subset of V, each pair of which is connected by an edge in E. Each clique is replaced by a star-shaped subgraph centered at a dummy node, which is shown as a circle in the abstract graph.
2. *Collapse a group of nodes with the same interactions into a composite node.* A group of nodes with the identical interacting partners are collapsed into a single composite node, which is shown as a diamond in the abstract graph.

A clique with n nodes contains $n(n - 1)/2$ edges, and a star-shaped graph for the clique contains exactly n edges. Therefore, replacing the cliques with star-shaped graphs substantially reduces the number of edges. Finding a clique with a maximum size in a graph is a NP-hard

*To whom correspondence should be addressed.

Algorithm 1 FindClique

```
1:  for all $v \in V$ do
2:    for all neighbor $u$ of $v$ do
3:      $Lst \leftarrow \emptyset$
4:      if not isClique($u$, $v$) then
5:        $Lst[0].node \leftarrow v$
6:        $Lst[0].idx \leftarrow v(u) + 1$
                                  {$v(u)$: index of $u$ in $v$}
7:        $Lst[1].node \leftarrow u$
8:        $Lst[1].idx \leftarrow u(v) + 1$
9:        repeat
10:         if not isClique($Lst[0].node$,
            $Lst[0].node[Lst[0].idx]$ then
11:           ChkClique($1, Lst[0].node[Lst[0].idx]$)
12:         end if
13:         $Lst[0].idx \leftarrow List[0].idx + 1$
14:       until ($Lst[0].idx \geq |Lst[0].node|$)
15:       if ($|Lst| \geq 4$) then
16:         DeclareClique($Lst$)
17:       end if
18:     end if
19:   end for
20: end for
```

problem (Battiti and Protasi, 2001). We have developed an efficient, heuristic algorithm that identifies all edge-disjoint cliques (see Algorithms 1 and 2). Collapsing is done for a clique of size 4 or more (line 15 in Algorithm 1) since collapsing a clique of size 3 does not decrease the number of edges. A description of identifying every group of nodes with the same interactions is given in Algorithm 3. While collapsing the cliques into a star-shaped graph only reduces the number of edges, collapsing the nodes with the same interactions into a composite node reduces the number of nodes as well as the number of edges.

InterViewer3 takes the input interaction data in several formats:

- data from a Microsoft Access database.
- graph modelling language (GML) format (http://www.uni-passau.de/Graphlet/GML).
- pnm: a pair of interacting protein names, separated by a space or tab, in each line.
- pid: a pair of interacting protein indices, separated by a tab, in each line.

As output, InterViewer3 produces two types of drawings (bitmap and GML file). The protein interaction data can also be saved in an ASCII file in any of the input formats described above.

Algorithm 2 ChkClique($N, NVal$)

```
if ($N = |Lst|$) then
  $Lst[|Lst|].node \leftarrow NVal$; $Lst[|Lst|].idx \leftarrow 0$
else
  if ($Lst[N].idx < |Lst[N].node|$) then
    if ($NVal > Lst[N].node[Lst[N].idx]$) then
      repeat
        $Lst[N].idx \leftarrow Lst[N].idx + 1$
      until ($NVal \leq Lst[N].node[Lst[N].idx]$)
      if ($NVal = Lst[N].node[Lst[N].idx]$)
      then
        ChkClique($N + 1, NVal$)
      end if
    end if
  end if
end if
```

Algorithm 3 MakeCompositeNode

```
for all $v \in V$ do
  if $v \notin$ CompositeNode then
    $CList \leftarrow \emptyset$
    for all $v'$ such that degree($v'$)=degree($v$) do
      if $v' \notin$ CompositeNode and
      $v$ and $v'$ have a same set of edges then
        Add $v'$ to $CList$
      end if
    end for
    MakeComposite($CList$)
  end if
end for
```

InterViewer3 was implemented in Borland Delphi 6.0, and runs on a Windows system. Figures 1 and 2 show examples of simplifying protein interaction networks using the two collapsing operations. For a graph with many cliques, the first collapsing operation alone is effective in reducing the complexity of a graph, as shown in Figure 2b. For a graph with few cliques, applying both collapsing operations is more effective in reducing the complexity, as shown in Figures 1c and 2c. A simplified graph can be expanded to a detailed graph as required.

For the purpose of comparing the actual running times of our algorithm with others, two other graph-drawing programs, Pajek (Batagelj and Mrvar, 2001) and Tulip (David, 2001), were run. Table 1 shows the running times of the following 5 layout algorithms on the same set of test cases: the new algorithm in InterViewer3, Kamada and Kawai's layout (1989) in Pajek, Fruchterman and Reingold's layout (1991) in Pajek, the GEM layout in Pajek, and the Spring-Electric force layout in Tulip. Pajek with Kamada and Kawai's layout algorithm could not

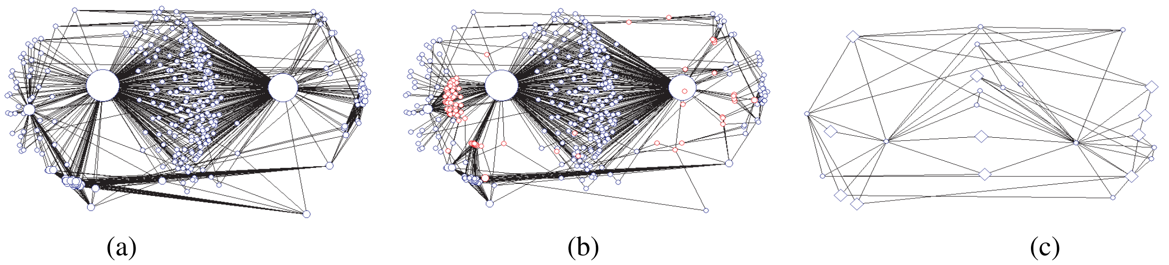

Fig. 1. (a) A protein interaction network with 307 nodes and 1063 edges. (b) Simplified graph with 332 nodes and 712 edges by replacing the cliques in Figure 1a with star-shaped subgraphs centered at dummy nodes, shown as red circles. (c) Simplified graph with 25 nodes and 62 edges by collapsing a group of nodes with the same interacting partners in Figure 1a into composite nodes, shown as diamonds.

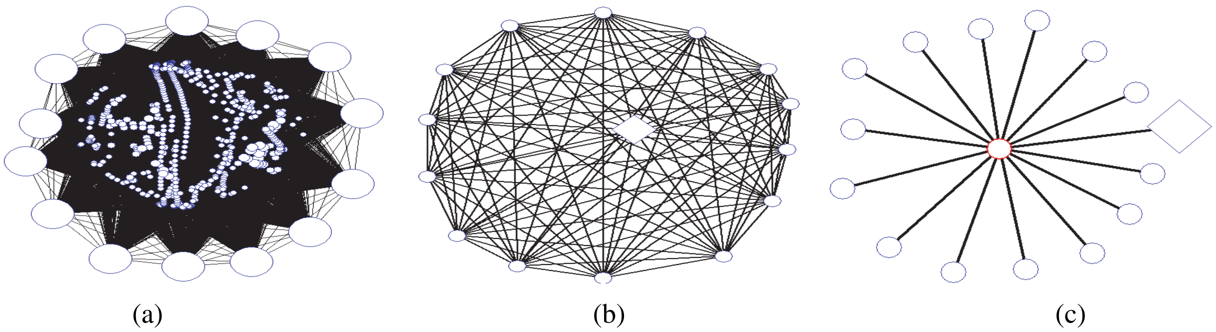

Fig. 2. (a) A protein interaction network with 527 nodes and 8568 edges. (b) Simplified graph with 15 nodes and 105 edges by collapsing a group of nodes in Figure 2a into composite nodes. (c) Simplified graph with 16 nodes and 15 edges by replacing a clique of Figure 2b with a star-shaped subgraph.

Table 1. Running times of the graph drawing programs on the 5 test cases on a Pentium IV 2.0GHz processor with 1GB memory

program (layout algorithm)	Y2H data (3751 nodes, 12917 edges)	BIND data (4048 nodes, 8286 edges)	DIP data (4690 nodes, 14460 edges)	human map 1 (8654 nodes, 184407 edges)	human map 2 (12056 nodes, 6989558 edges)
InterViewer3	7 sec	6 sec	7 sec	19 sec	8 min 20 sec
Pajek (K-K)	2 min 31 sec	1 min 37 sec	3 min 04 sec	56 min 38 sec	out of memory
Pajek (F-R)	28 min 23 sec	20 min 02 sec	42 min 45 sec	2 hours 28 min 40 sec	5 hours 43 min 32 sec
Tulip (GEM)	2 min 10 sec	4 min 40 sec	18 min 40 sec	9 hours 19 min 10 sec	≫ 10 hours
Tulip (S-E)	24 min 35 sec	35 min 47 sec	56 min 45 sec	≫ 10 hours	≫ 10 hours

K-K: Kamada-Kawai's layout, F-R: Frunchterman-Reingold's layout, S-E: Spring-Electric force layout, human map 1 & 2: human protein interaction data

even visualize the human map 2 data due to an 'out of memory' error. It follows from this comparison that InterViewer3 is at least one order of magnitude faster than the recent implementations of the force-directed layout algorithms.

ACKNOWLEDGEMENTS

This work was supported by the Ministry of Information and Communication of Korea under grant number IMT2000-C3-4.

REFERENCES

Batagelj,V. and Mrvar,A. (2001) Pajek–analysis and visualization of large networks. *LNCS*, **2265**, 477–478.

Battiti,R. and Protasi,M. (2001) Reactive local search for the maximum clique problem. *Algorithmica*, **29**, 610–637.

David,A. (2001) Tulip. *LNCS*, **2265**, 435–437.

Fruchterman,T.M.J. and Reingold,E.M. (1991) Graph drawing by force-directed placement. *Software-Practice and Experience*, **21**, 1129–1164.

Ju,B.-H., Park,B., Park,J.H. and Han,K. (2003) Visualization and analysis of protein interactions. *Bioinformatics*, **19**, 317–318.

Kamada,T. and Kawai,S. (1991) An algorithm for drawing general undirected graphs. *Information Processing Letters*, **31**, 7–15.