IET Systems Biology

Special Issue: Computational Translational Medicine

IET Journals
The Institution of Engineering and Technology

Graphics processing unit-based alignment of protein interaction networks

ISSN 1751-8849
Received on 5th December 2014
Revised on 23rd January 2015
Accepted on 3rd March 2015
doi: 10.1049/iet-syb.2014.0052
www.ietdl.org

Jiang Xie[1] ✉, *Zhonghua Zhou*[1], *Jin Ma*[1], *Chaojuan Xiang*[1], *Qing Nie*[1,2], *Wu Zhang*[1]

[1]*School of Computer Engineering and Science, Shanghai University, Shanghai, People's Republic of China*
[2]*Department of Mathematics, Center for Mathematical and Computational Biology, University of California at Irvine, California, USA*
✉ *E-mail: jiangx@shu.edu.cn*

Abstract: Network alignment is an important bridge to understanding human protein–protein interactions (PPIs) and functions through model organisms. However, the underlying subgraph isomorphism problem complicates and increases the time required to align protein interaction networks (PINs). Parallel computing technology is an effective solution to the challenge of aligning large-scale networks via sequential computing. In this study, the typical Hungarian-Greedy Algorithm (HGA) is used as an example for PIN alignment. The authors propose a HGA with 2-nearest neighbours (HGA-2N) and implement its graphics processing unit (GPU) acceleration. Numerical experiments demonstrate that HGA-2N can find alignments that are close to those found by HGA while dramatically reducing computing time. The GPU implementation of HGA-2N optimises the parallel pattern, computing mode and storage mode and it improves the computing time ratio between the CPU and GPU compared with HGA when large-scale networks are considered. By using HGA-2N in GPUs, conserved PPIs can be observed, and potential PPIs can be predicted. Among the predictions based on 25 common Gene Ontology terms, 42.8% can be found in the Human Protein Reference Database. Furthermore, a new method of reconstructing phylogenetic trees is introduced, which shows the same relationships among five herpes viruses that are obtained using other methods.

1 Introduction

Biomolecular network alignment is an effective approach for understanding similarities and dissimilarities between different living systems. Furthermore, protein interaction network (PIN) alignment facilitates the exploration of protein–protein interactions (PPIs), the prediction of protein functions and the study of evolution. There are two general classes of PIN alignment algorithms: local and global. Local alignment is ambiguous because one node in a network can be matched with numerous nodes in another network. Sub-networks of PINs, including pathways and protein complexes, can be identified using this method. PathBLAST [1] was one of the earliest algorithms developed for local PIN alignment, and NetworkBLAST-M [2] is a modified form of this algorithm. In global network alignment, each node from one network is matched with a unique node in another network. Berger's group proposed the first global PIN alignment algorithm, IsoRank [3], and further improved the algorithm to generate IsoRankN [4] and PISwap [5]. Natasa's group proposed a series of algorithms: GRAAL [6], H-GRAAL [7] and MI-GRAAL [8]. Based on the IsoRank algorithm, Andrei *et al.* developed an alignment method for probabilistic PINs [9]. GEDEVO [10] is an ingenious method that aligns networks using a novel evolutionary algorithm and attempts to minimise the GED. NETAL [11] deals with topological and biological scores separately, then uses GA to find the global alignment. NETAL focuses on topological similarities in the current version and aligns networks very quickly. The Hungarian-Greedy Algorithm (HGA) [12], which is improved from the Immediate Neighbours-in-first Method (INM) [13], is an adaptive hybrid algorithm for global network alignment. INM has many of the common features of PIN alignment algorithms, such as considering the attributes of proteins and interactions between them, computing the similarity matrix between the aligned networks, and calling attention to the neighbours of each protein.

The underlying subgraph isomorphism problem is referred to as NP-hard, and it complicates and increases the time required to align PINs. Moreover, increasing amounts of biological data, such as the data generated through completion of the Human Genome Project (HGP) and the Encyclopaedia of DNA Elements project, will rapidly increase the size of these data sets [14]. The exploration of larger PINs based on large sets of data requires improved alignment algorithms and adaptation of existing sequential algorithms for parallel simulations.

The graphics processing units (GPUs) of many cores are multi-threaded chips capable of hundreds of trillions of peak floating point operations per second (FLOPS). Although these chips were originally designed to accelerate graphics processing, the exceptional performance-to-cost ratio of GPUs has enabled supercomputing power to be achieved in desktop units. The application of GPUs was hindered by the complexity of implementation until Compute Unified Device Architecture (CUDA) was introduced by NVIDIA [15] in 2007. In recent years, GPUs have been widely used in high-performance computing and are exploited for running numerous bioinformatics algorithms, such as the Smith-Waterman alignment algorithm [16], molecular docking [17], BLAST [18], and network clustering [19]. Most of these applications have concentrated on single-molecule systems, and attempts to align PINs have been limited.

Since only highly parallelised algorithms can be efficiently run on GPUs as a result of their single instruction multiple data (SIMD) architecture, it is difficult to improve alignment results because network topological information is frequently used in PIN alignment algorithms. The high data correlation that occurs in typical PIN alignment algorithms limits the advantages of GPUs. Within the framework of NVIDIA's GPUs, we propose several methods of addressing these difficulties in PIN alignment using GPUs. Using HGA as an example, we have improved the algorithm by creating an HGA with 2-nearest neighbours (HGA-2N) and implementing its GPU acceleration. The remainder of this paper is organised as follows. In Section 2, HGA, time complexity and HGA-2N are illustrated. In Section 3, the acceleration methods for PIN alignment using GPUs are presented. In Section 4, numerical experiments are designed to analyse

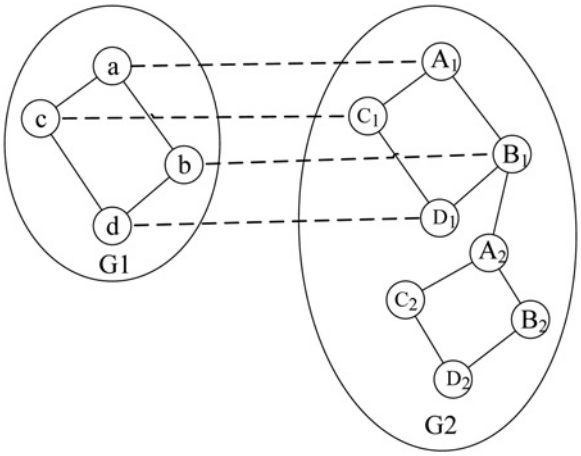

Fig. 1 *Example of PIN alignment*

Nodes connected by dotted lines in the figure are the matched nodes, and this alignment provides the best mapping of the two networks

Table 1 Similarities between nodes from two different networks

	A_1	B_1	C_1	D_1	A_2	B_2	C_2	D_2
a	0.8	0	0.1	0.1	0.5	0	0	0
b	0.5	0.7	0	0.2	0	0	0.1	0
c	0.2	0	0.9	0	0.3	0.4	0	0
d	0.1	0	0.1	0.5	0.2	0	0.1	0

computing time and explore the computing time ratio between the CPU and GPU (rct-CG). In addition, the running time and solution quality of HGA, HGA-2N and some popular algorithms are compared. Furthermore, we study the alignments obtained with HGA and HGA-2N and reconstruct the phylogenetic tree of five herpes viruses. In Section 5, our conclusions are presented.

2 PIN alignment

2.1 Problem definition

A PIN can be represented as an undirected and unweighted graph and denoted by $G(V, E)$, where V is a set of vertexes, and E is a set of edges representing PPIs. The alignment of PINs is defined below.

Given two PINs represented by the graphs $G_1(E_1, V_1)$ and $G_2(E_2, V_2)$, in which $|V_1| = n_1$ and $|V_2| = n_2$, alignment is conducted to determine the mapping, φ, between the proteins of the two networks that best represents conserved biological functions. This problem can be formulated as follows

$$\mathrm{Sim}(G_1, G_2) = \arg\max \sum_{a \in V_1} \mathrm{sim}(a, \varphi(a)) \tag{1}$$

in which $\mathrm{sim}(a, \varphi(a))$ represents the biological similarities between a in G_1 and the matching vertex $\varphi(a)$ in G_2, and the mapping, φ, shows the largest value in $\mathrm{Sim}(G_1, G_2)$. Fig. 1 presents an example of a PIN alignment, and Table 1 shows the similarities between the nodes from the two networks in Fig. 1.

In PIN alignment, the two graphs are represented by two adjacent matrices: $M_{1(n_1 \times n_1)}$ and $M_{2(n_2 \times n_2)}$. The entry $M_1(a, b)$ equals '1' if a, $b \in G_1$ are neighbours; otherwise, this entry is '0'. The diagonal elements in M_1 are all '0'. The entries in M_2 are the same as in M_1. Similarities are stored in another matrix, $\boldsymbol{S}_{(n_1 \times n_2)}$. Each entry, $S(u, v)$, indicates the similarity between $u \in G_1$ and $v \in G_2$.

2.2 Algorithm

The HGA is a typical PIN alignment algorithm, and it has two main stages. One stage involves the computation of similarities between the nodes from two networks in the similarity matrix, $\boldsymbol{S}$. In this stage, the similarities between the neighbours and non-neighbours of each pair of nodes are calculated [12]. The second stage involves mapping between nodes.

The pseudo code of HGA is as follows:

See Fig. 2.

In the above code, Step 4 and Step 5 account for most of the calculation time. Since G_1 has n_1 nodes and G_2 has n_2 nodes, in which $n_1 \le n_2$, the MateList^k has n_1 pairs of nodes. Given any two nodes, a and b, in the aligned networks, N_1 indicates the average similarity between their neighbours; N_2 indicates the average similarity between their non-neighbours; the time complexities of N_1 and N_2 are $O(n_1 \times n_2)$; and the time complexity of Steps 4 and 5 is $O(n_1^2 \times n_2)$ and $O((n_1 \times n_2)^2)$, respectively, with each loop equal to $O((n_1 \times n_2)^2)$. Increases in the size of the network will result in more computational time being required for this algorithm.

In HGA, N_1 and N_2 are computed as follows

$$N_1^{(k)}(a, b) = \begin{cases} \sum\limits_{a_2 \leftrightarrow a, b_2 \leftrightarrow b} \dfrac{S^{(k)}(a_2, b_2)}{d(a) \times d(b)}, & \text{if } (d(a) \neq 0, d(b) \neq 0) \\ \sum\limits_{a_2 \in V_1, b_2 \in V_2} \dfrac{S^{(k)}(a_2, b_2)}{n_1 \times n_2}, & \text{if } (d(a) = d(b) = 0) \\ 0, & \text{others} \end{cases} \tag{2}$$

(see (3))

As typical sparse networks, the average degree of PINs may be <8, meaning that each node has a greater number of non-neighbours than neighbours. Calculating the similarities between all non-neighbours in such cases is costly. Proteins, such as protein a, always interact directly with proteins that have similar functions. Therefore the proteins that are closer to a have more of an effect on the function of a. Thus, for the non-neighbours of the computed nodes, the influence of nodes that are far away (i.e. distance ≥3) from the computed nodes is weak, whereas the influence of the neighbours' neighbours of the computed nodes is significant. To consider these factors, we improved HGA and developed HGA-2N, in which the average similarity of non-neighbours between nodes a and b is computed using the 2-nearest neighbours, rather than all of the non-neighbours. The formulas for calculating the new N_1 and N_2 are as follows

$$N_1^{(k)}(a, b)' = \begin{cases} \sum\limits_{a_2 \leftrightarrow a, b_2 \leftrightarrow b} \dfrac{S^{(k)}(a_2, b_2)}{d(a) \times d(b)}, & \text{if } (d(a) \neq 0, d(b) \neq 0) \\ 0, & \text{others} \end{cases} \tag{4}$$

$$N_2^{(k)}(a, b) = \begin{cases} \sum\limits_{a_2 \leftrightarrow a, b_2 \leftrightarrow b} \dfrac{S^{(k)}(a_2, b_2)}{(n_1 - d(a)) \times (n_2 - d(b))}, & \text{if } (d(a) \neq n_1 - 1, d(b) \neq n_2 - 1) \\ \sum\limits_{a_2 \in V_1, b_2 \in V_2} \dfrac{S^{(k)}(a_2, b_2)}{n_1 \times n_2}, & \text{if } (n_1 - d(a) = n_2 - d(b) = 1) \\ 0, & \text{others} \end{cases} \tag{3}$$

```
Algorithm1 HGA(G1, G2, S0)
{
  //Step 1: Obtain initial similarity matrix
  S0=BLAST(G1, G2);
  Sk=S0;
  while(1)
  {
// Step 2: Obtain match results from the similarity matrix
MateListk=match(G1, G2, Sk);
// Step 3: Calculate the score from the match results
SSk,ECk=mark(MateListk);
Scorek=SSk+100*ECk
// Step 4: Update the similarity matrix
for(i=MateListk.Begin;i<MateListk.End;i++)
{
  u=i.OneVertex;
  v=i.AnotherVertex;
  for(j=G1.Begin;j<G1.End;j++)
    for(m=G2.Begin;m<G2.End;m++)
    {
      if((u and j are neighbours) && (v and m are neighbours))
      {
       deltaSim=Sk[u,v]/Deg(j);
       NewSk[j,m]+=deltaSim;
      }
    }
}
// Step 5: Iterate and calculate the next similarity matrix
for(i=1; i<n1; i++)
  for(j=1; j<n2; j++)
  {
    N1(i,j) = ComputeN1( );
    N2(i,j) = ComputeN2( );
    Sk+1(i,j) = (1-α)S0(i,j)+α(N1(i,j)+N2(i,j))/2;
  }
// Step 6: Verify the convergence of the results
if(Scorek=Scorek+1=Scorek-1){ break;}
    if(max|Sk-Sk-1|<0.01){ break;}
    if(max|Sk+1-Sk|<0.01){ break;}
    Sk=Sk+1;
  }
}
```

Fig. 2 *Algorithm 1: HGA(G_1, G_2, S^0)*

$$N_2^{(k)}(a,b)' = \begin{cases} \sum_{a_2 \Leftrightarrow a, b_2 \Leftrightarrow b} \dfrac{S^{(k)}(a_2,b_2)}{d^{2\prime}(a) \times d^{2\prime}(b)}, & \text{if } (d^2(a) \neq 0, d^2(b) \neq 0) \\ 0, & \text{others} \end{cases} \tag{5}$$

where k indicates the iteration time; $d(a)$ indicates the degree of a; $d^2(a)$ indicates the number of 2-nearest neighbours of a; $a_2 \leftrightarrow a$ indicates that a_2 is the neighbour of a; and $a \Leftrightarrow a_2$ indicates that a_2 is a 2-nearest neighbour of a. Except for the computation of N_1 and N_2, the pseudo code of HGA-2N is same as that of HGA.

In addition to edge correctness (EC) [8], which is often used to compute the percentage of matched edges, we employ the sum score (SS) [12] to measure the alignments. SS is a factor that computes sequence and topology similarities. Experiments have demonstrated that the alignments obtained with HGA-2N are close to those obtained with HGA, which is described in Section 4.

3 Architecture and implementation

Owing to the global nature of the algorithms employed for PIN alignment, implementation using GPUs involves a number of general difficulties in parallelisation and communication. Our approaches for resolving certain difficulties are described below.

3.1 Parallel pattern

The first challenge is the parallel computing pattern, which is used to allocate $\boldsymbol{S}$.

We propose a pattern of parallel computing for processing $\boldsymbol{S}$ based on two main characteristics of the architecture of the threads of GPUs. First, threads can be conveniently controlled in GPUs. Second, compared with CPUs, the number of processors in GPUs is greater, and the computational power of each processor is relatively lower. This pattern leads to a high grain and efficiency in the parallel computing pattern.

The method used to compute the unique ID of each entry is as follows

$$\text{EntryID}^{S(i,j)} = i \times n_2 + j \tag{6}$$

The next question is how to compute the ID of each thread. Fig. 3*a* illustrates the framework of the threads in typical GPUs. The ID of each thread can be computed using the following formula

$$\text{ThreadID} = \text{BlockId}X.X \times \text{BlockDim} + \text{ThreadId}X.X \tag{7}$$

where ThreadId$x{\cdot}x$ indicates the number of threads in the same block, and BlockId$x{\cdot}x$ denotes the number of blocks in the grid.

Table 2 presents the correspondence between the IDs of the threads and the entries. A thread computes the entry of the same ID. In this example, there are 32 threads in each block.

3.2 Computing mode

The processors in GPUs are powerful for performing calculations but are not efficient for making logical decisions. Each thread typically requires its neighbours' information to complete the calculation. Thus, each thread must traverse M_1 and M_2 to determine whether the nodes are the neighbours of the computing nodes. Large numbers of logical decisions reduce the performance of GPU programs. To reduce the frequency of such judgments, we have built two new matrices (NM_1 and NM_2) by traversing the two adjacent matrices. The size of these matrices is identical to the size of M_1 and M_2. Each line in these matrices consists of three components of topological information related to one node. The first component is one entry for the number of neighbours of the node; the second component contains all of the neighbours of the node; and the third component contains all of the non-neighbours

IET Syst. Biol., 2015, Vol. 9, Iss. 4, pp. 120–127

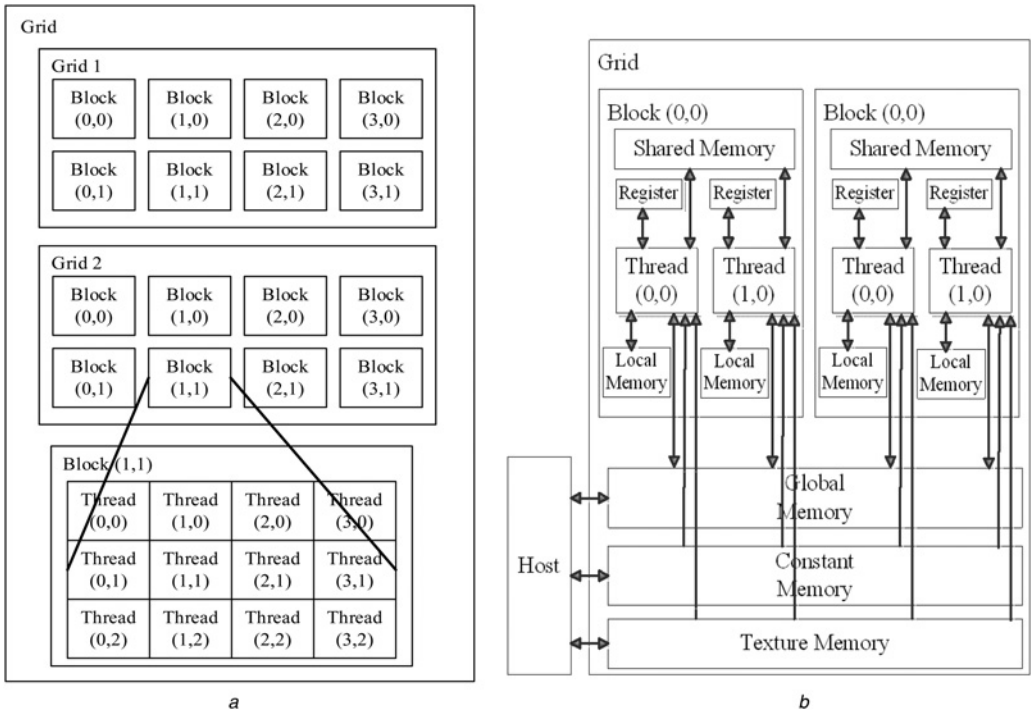

Fig. 3 *Architecture of GPU*

a Model of the structure of threads
b Models of memory in GPUs

Table 2 Correspondence between the ThreadID and EntryID

BlockId*x.x*	0	1	2	…	BlockDim
ThreadId*x.x*	0 1 2 … 31	0 1 2 … 31	0 1 2 … 31	…	0 1 2 …(n_1*n_2)%32
ThreadID	0 1 2 … 31	32 33 34 … 63	64 65 66 … 95	BlockId*x.x**BlockDi*m*+ThreadId*x.x*	… n_1*n_{2-1}
EntryID	0 1 2 … 31	32 33 34 … 63	64 65 66 … 95	(*i**n_2+*j*) …	… n_1*n_{2-1}
Entry	(0,0) (0,1) (0,2) …	…	…	(*i*, *j*) …	… (n_{1-1}, n_{2-1})

of the node. Fig. 4 illustrates the construction process for the new matrices. The neighbours of each node can be obtained directly based on the new matrix, and the judgment times and frequency of memory access are reduced.

3.3 Storage mode

Different types of memory are associated with different sizes, speeds and functionalities in GPUs. Fig. 3*b* illustrates the memory architecture of GPUs. The global memory is the largest in size and slowest in speed, and there are a maximum number of threads that can be physically executed in parallel; this group of threads is known as a warp. Constant memory can provide a higher bandwidth than global memory if all of the threads of a half warp uses the same input data. Otherwise, the performance may be reduced. Shared memory is employed for data communication between threads that belong to the same block. If one variable is shared, every block will have a copy and will be shared by all of the threads in one block. Shared memory is the fastest type of memory, but it is the smallest and is most difficult to extend. GPUs also have local memory, texture memory and registers, which are difficult to control.

As noted above, when computing one pair of nodes (i.e. one entry in ***S***), information related to the neighbours of those nodes, such as similarities and matching statuses, is used. In addition, all of the threads compute different pairs of nodes simultaneously and require different information. Thus, the three matrices (M_1, M_2 and ***S***) should be transmitted to GPUs prior to computing and cannot be saved in a distributed manner.

Global memory is the best type of memory for PIN alignment. Although the global memory is the largest in size, even this type of memory is not sufficiently large to save these three matrices because the scale of PINs is too large. We address this problem using two approaches: one employs a triangular matrix to save the two symmetrical adjacent matrices, and the other employs 'unsigned char' to define entries in two adjacent matrices instead of using 'int' because entries in these matrices only equal '0' or

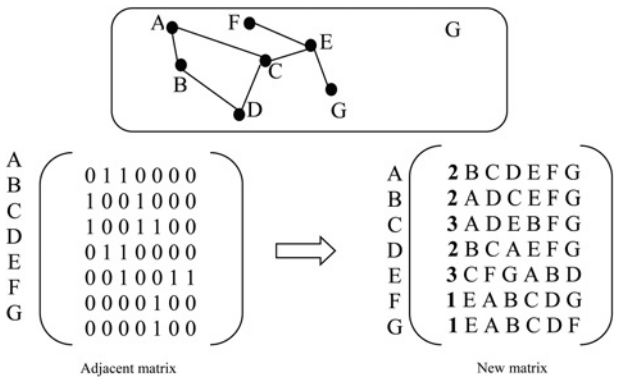

Fig. 4 *Construction process for the new matrix*

For example, in graph G, two nodes, 'C' and 'B' are neighbours of node 'A'; thus, in line 1 of the new matrix, the first entry is 2; the next two entries are 'C' and 'B'; and the next four are 'D', 'E', 'F' and 'G', which are the non-neighbours of 'A'

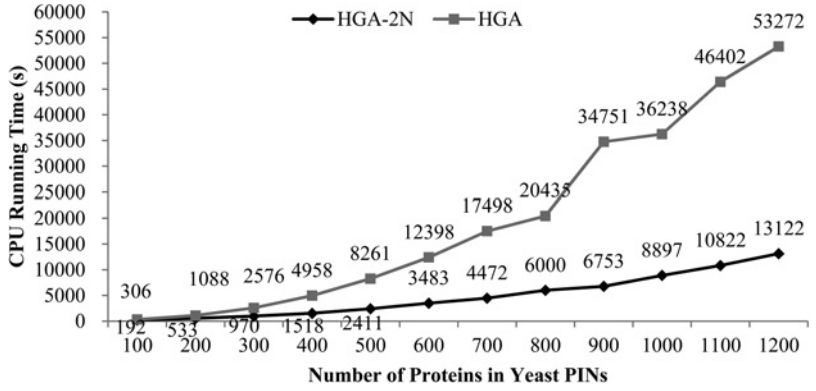

Fig. 5 *Average CPU running time of HGA and HGA-2N on each group*

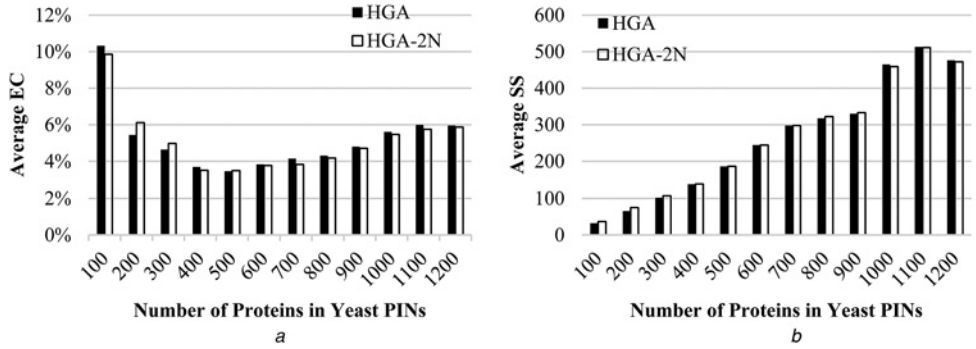

Fig. 6 *Results of HGA-2N and HGA for data set 1*

a Average ECs
b Average SSs

'1'. By combining these two methods, less than one-eighth of the memory required by the regular mode is used.

4 Experiments

4.1 Platform

The GPU devices employed in this study have two Intel(R) Xeon(R) E5-2680 CPUs and two NVIDIA Tesla M2090s on ZQ 4000 located at Shanghai University. Each CPU device has 8 cores, while each NVIDIA device has 512 cores, and the peak double-precision floating point reaches 665 GFlops. The software environment is CUDA-5.0 in CentOS Linux 5.7.

The CPU program is run on a DELL Poweredge T110. The computer has 1 processor that has 8 2.53 GHz Intel Xeon X3440 cores with 8 G memory. The software environment is gcc 4.9.2 in CentOS 5.6.

4.2 Data sets

There are two data sets in our experiments. Data set 1 is for the yeast and human PINs [6]. G_{1s} represent the connected subnetworks extracted randomly from the yeast PIN, and the number of nodes in the subnetworks increases from 100 to 1200 in increments of 100. For each increment, 50 different G_{1s} with same number of nodes are produced and aligned withG_2, and the final alignment is the average of these 50 alignments. G_2 has 4451 nodes and 11 096 edges, and it is a connected subnetwork randomly extracted from the human PIN.

Data set 2 includes three pairs of PINs from different species: yeast and human [6], *Campylobacter jejuni* (*C. Jejuni*) and *Escherichia coli* (*E. Coli*) [6], and *Caenorhabditis elegans* (*C. Elegans*) and *Drosophila melanogaster* (*D. Melanogaster*) [5]. The details of these PINs are shown in Table 5.

4.3 Computing time

In this section, we compare the CPU computing time (sequential program) of HGA and HGA-2N using data set 1. Fig. 5 displays the average running time of HGA and HGA-2N for each group, which indicates that HGA-2N runs much faster than HGA especially for larger networks.

4.4 Alignment results

First, the alignments obtained for data set 1 using HGA-2N and HGA are compared. Fig. 6 shows the ECs and SSs. The

Table 3 Comparison of large-scale PIN alignment results and *p*-values

	Algorithm	*C. Jejuni–E. Coli*	Yeast-Human	*C. Elegans-D. Melanogaster*
SS	HGA	75.86	628.18	57.96
	HGA-2N	76.74	633.16	57.96
	difference	1.1%	0.8%	0
EC	HGA	10.11%	7.14%	18.2%
	HGA-2N	10.14%	7.15%	19.2%
	difference	0.03%	0.01%	1.0%
p-value	HGA-2N	8.4×10^{-5}	4.9×10^{-2}	4.0×10^{-3}

IET Syst. Biol., 2015, Vol. 9, Iss. 4, pp. 120–127

Table 4 Average percentage of the time spent computing Step 5 for networks of different sizes

number of proteins in G_1	100	200	300	400	500	600	700	800	900	1000	1100	1200
time percentage of HGA (%)	40.2	67.0	85.0	92.2	95.0	96.9	97.7	98.0	98.5	98.7	98.8	98.9
time percentage of HGA-2N (%)	27.6	35.2	58.0	74.5	82.8	87.8	90.9	92.5	93.8	94.7	95.2	95.4

Table 5 Scales and rct-CGs of large-scale PINs

Species	C.J	E.C	Yeast	Human	C.E	D.M
number of nodes	1,111	1,941	2,390	4,451	2,742	6,697
number of edges	2,988	3,989	16,127	11,096	5,851	26,712
CPU run times (s)	1,194		30,076		33,766	
GPU run times (s)	44		203		225	
rct-CGs	27.14		148.15		150.07	

results shown in Fig. 6 indicate that both the ECs and SSs of HGA-2N are close to those of HGA for different sizes of aligned networks.

Next, the alignments for data set 2 are compared, as shown in Table 3. The alignments are close to each other, and the differences between them range from 0 to 1.1%. For each pair of PINs, we randomly change the connections in one PIN, make 50 different random networks and align them with another PIN to compute *p*-values [9]. The *p*-values of the three pairs of PINs shown in Table 3 are all <0.01, indicating that the alignments exhibit high statistical significance.

4.5 Computing time ratio between CPU and GPU

As previously mentioned, computing the similarity matrix in Step 5 is time consuming. Based on data set 1, Table 4 shows the average percentage of the time spent computing Step 5 of HGA and HGA-2N, respectively.

According to the methods described in Section 3, the parallelisation of step 5 is realised using GPUs. The running time of the GPU program is then tested and compared with the CPU implementation time. Here, we define the computing time ratio between CPU and GPU (rct-CG) as

$$\text{rct-CG} = \frac{\text{computing time of CPU}}{\text{computing time of GPU}} \tag{8}$$

For data set 1, the average rct-CG is computed and presented in Fig. 7, which demonstrates that the rct-CG of HGA-2N increases along with the increases in the size of the alignment networks, indicating that our implementation is effective for large-scale PIN alignment. However, the rct-CG of HGA decreases when the number of nodes in G_1 increases.

Furthermore, data set 2 is used to compute the rct-CGs of large-scale PINs. Table 5 shows the scales and rct-CGs of data set 2, which indicate that the rct-CGs for large-scale PIN alignments are indeed high. In addition, the rct-CG improves along with increases in the size of the PINs, and a 150-fold rct-CG can be achieved.

Table 6 Comparison between HGA-2N and other algorithms for yeast and human PINs

Algorithm	SS	EC (%)	HGp	Parameters
IsoRank	333.92	3.63	663	default
GRAAL	0	3.87	5	default
MI-GRAAL	238.52	11.86	387	*SeqD*
NETAL	0	21.44	5	$a = 0.0001$, $i = 2$
PISwap	698.59	2.73	1204	default
GEDEVO	53.45	10.52	86	maxsame = 5000
HGA	628.18	7.14	1258	$\alpha = 0.4$
HGA-2N	**633.16**	**7.15**	**1246**	$\boldsymbol{\alpha = 0.4}$

4.6 Comparison of other algorithms

There are many algorithms for network alignment, and some of them have high performance. Here, we compare HGA, HGA-2N with some popular algorithms based on the yeast and human PINs of data set 2, as shown in Tables 6 and 7.

Table 6 list their SSs, ECs and HGps, where HGp is the number of HomoloGene pairs among matched proteins. The more HGps found by one algorithm, the more homologous proteins are matched. Thus, the algorithm can find the alignment that expresses the more similarities between the aligned networks. The results shown in Table 6 indicate that HGA and HGA-2N can find the alignment with much more HGps. Although their SSs are less than PISwap, the ECs are higher.

Furthermore, we compare the running times of these algorithms. The sequential programs run on an HP Z800. This computer has 8 processors with 8 G memory, and each processor has 4 2.13 GHz Intel Xeon E5506 cores. The software environment is gcc 4.4.7 in Centos release 6.6(final). Table 7 indicates that HGA-2N run on the GPU dramatically reduces the computing time when large scale PINs are aligned.

4.7 PPI analyses

Gene Ontology (GO) terms annotate protein functions. If two nodes exhibit common GO terms, these nodes may present the same

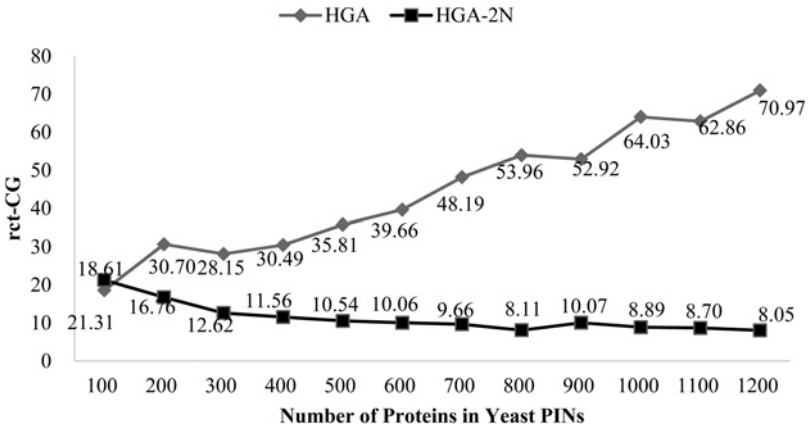

Fig. 7 *Average rct-CG of HGA-2N and HGA*

Table 7 Running time of HGA-2N and other algorithms for yeast and human PINs

Algorithm	IsoRank	GRAAL	MI-GRAAL	NETAL	PISwap	GEDEVO	HGA	**HGA-2N**
sequential computing time	15min11s	46min8s	1h14min48s	48s	5min1s	>3days	>3days	**~8h**
parallel computing time	–	–	–	–	–	13h23min08s	6h58min56s	**3min23s**
parameters	default	default	*SeqD*	$a = 0.0001$, $i = 2$	default	*maxsame* = 5000	$\alpha = 0.4$	**$\alpha = 0.4$**
remark	–	–	–	–	–	16 CPU cores on GPU	GPU	**GPU**

Table 8 Interactions in the human and yeast PINs

		1CGOP	5CGOPs	10CGOPs	15CGOPs	20CGOPs	25CGOPs
conserved interactions		1,245	159	58	30	12	6
yeast	predicted interactions	1,617	413	101	32	10	6
human	predicted interactions	3,882	696	185	61	18	7
	existing in HPRD	–	–	34	26	7	3
	to be proved	–	–	151	35	11	4

Table 9 Human PPIs predicted by 25CGOPs

	Interactions	Interactions
to be proved	LAS17 SLA1	PRP19 SYF1
	RPT1 RPT3	TAF6 TRA1
existing in HPRD	TAF10 TAF6	TAF1 TAF10
	TAF1 TAF6	

functions. For alignment, if one pair of matched nodes presents common GO terms, the pair is referred to as a Common Gene Ontology Pair (CGOP); 1CGOP indicates that the nodes have at least one common GO, while 5CGOPs indicates that the nodes have at least five GOs and so on.

Proteins usually interact and work together with other proteins. Given two CGOPs, u, $\varphi(u)$ and v, $\varphi(v)$, there is an interaction between u and v. If there is an interaction between $\varphi(u)$ and $\varphi(v)$,

Table 10 Alignment results for five viruses

pairs of viruses	EBV KSHV	mCMV EBV	HSV-1 VZV	mCMV HSV-1	EBV VZV
SS	25.1	18.4	17.2	13.7	12.5
EC	26%	17%	27%	21%	13%
pairs of viruses	EBV HSV-1	KSHV VZV	mCMV KSHV	mCMV VZV	HSV-1 KSHV
SS	11.8	11.3	10.3	9.6	7.9
EC	21%	18%	12%	13%	9%

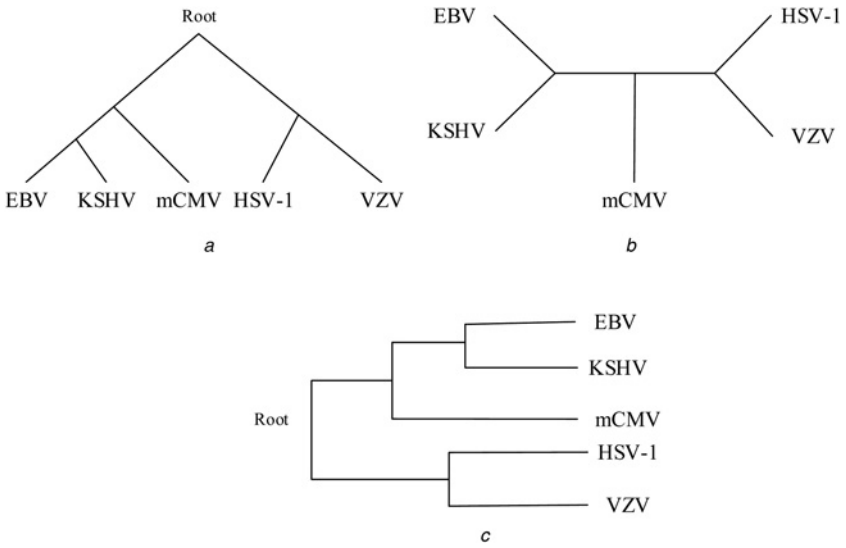

Fig. 8 *Phylogenetic trees reconstructed using three different methods*

a Phylogenetic tree reconstructed using the method of Fossum *et al.* [20]
b Phylogenetic tree reconstructed using the method of Natasa *et al.* [8]
c Phylogenetic tree reconstructed using HGA-2N
The three trees show the same phylogenetic relationships

IET Syst. Biol., 2015, Vol. 9, Iss. 4, pp. 120–127

these two interactions are conserved interactions. If there is no interaction between $\varphi(u)$ and $\varphi(v)$, a potential interaction can be predicted between $\varphi(u)$ and $\varphi(v)$. Based on different numbers of CGOPs in the alignment of human and yeast PINs, conserved interactions can be found, and new interactions in human and yeast PINs can be predicted. As shown in Table 8, 34 of 185 human PPIs, 26 of 61 human PPIs, 7 of 18 human PPIs, and 3 of 7 human PPIs predicted by 10 to 25 CGOPs can be found in the Human Protein Reference Database (HPRD); the percentage ranges from 18.4 to 42.8%, demonstrating the validity of our method. However, further studies must be performed for the remaining predicted interactions. The details of the predicted results of the 25CGOPs are shown in Table 9, and additional details of the predictions can be found in the supplementary file.

4.8 Phylogenetic tree

Fossum *et al.* [20] reconstructed phylogenetic relationships by counting the number of conserved interacting orthologous pairs in species, with a focus on biological relevance. Natasa and Przulj [8] used EC as the distance between species to reconstruct phylogenetic relationships, with a focus on topological structures. We present a new method for reconstructing phylogenetic trees that uses SS, which combines the biological sequence similarities and topological structures of PINs. First, the FASTA data for five herpes viruses are downloaded from the National Center for Biotechnology Information and Universal Protein Resource (Uniprot) databases, and then BLAST [21] is used to calculate their similarity. These viruses include varicella zoster virus (VZV), Kaposi's sarcoma-associated herpes virus (KSHV), herpes simplex virus 1 (HSV-1), murine cytomegalovirus (mCMV) and Epstein–Barr virus (EBV), and they are aligned with HGA-2N. Based on the alignment results (Table 10), a phylogenetic tree is reconstructed, which is shown in Fig. 8. The phylogenetic trees reconstructed using the three methods show the same phylogenetic relationships.

5 Conclusions

In this study, the typical algorithm HGA is used as an example for PIN alignment, and HGA-2N is proposed, followed by implementation of its GPU acceleration. This process utilised the architecture of GPUs and features of PIN alignment algorithms. The programs and alignments can be found at http://biocenter.shu.edu.cn/software/index.php/hga.

HGA-2N considers 2-nearest neighbours, rather than all non-neighbours, which reduces the scale of computing and balances tasks during parallelisation. The GPU implementation of HGA-2N optimises the parallel pattern, computing mode and storage mode. For the parallel pattern, a one-to-one correspondence between the threads in the GPUs and entries in the matrix is explored to enable full utilisation of the thread architecture in the GPUs. For the computing mode, an algorithm is introduced to reduce judgment times and allow full use of the GPU's calculating power. For the storage mode, we utilise an upper triangular matrix and 'unsigned char' to save memory. By integrating the three methods, the parallel algorithm of HGA-2N can achieve up to a 150-fold rct-CG in large-scale PIN alignment.

Numerical experiments have demonstrated that HGA-2N can find alignments that are close to those found by HGA while dramatically reducing computing time. Compared with other algorithms for PIN alignment, HGA-2N has better performance on HGp, EC and SS. By using GO terms for proteins and the results aligned with HGA-2N in GPUs, conserved interactions can be observed, and interactions in yeast and human PINs can be predicted. Among the predictions based on 10 to 25 common GO terms, 18.4% to 42.8% can be found in the HPRD [22], which indicates the validity of our predictions. Further analyses will be required for the remaining predicted interactions.

In addition, we introduce a new method of reconstructing phylogenetic trees using the significance of the biological relevance and topological structures of the alignment results. Five herpes virus PINs are aligned using the GPU algorithm, and their phylogenetic tree is reconstructed, revealing the same relationships shown using the previous methods.

In general, the new GPU alignment algorithm proposed in this paper provides many insights into the development of improved GPU implementations for networks alignment that takes both nodes and edges into consideration.

6 Acknowledgements

This work was supported by the Major Research Plan of NSFC (grant no. 91330116), Key Project of the Science and Technology Commission of Shanghai Municipality (grant no. 11510500300) and Specialized Research Fund for the Doctoral Programme of Higher Education (grant no. 20113108120022). Qing Nie would like to acknowledge support from the NSF and NIH.

7 References

1 Kelley, B.P., Yuan, B., Lewitter, F., *et al.*: 'PathBLAST: a tool for alignment of protein interaction networks', *Nucleic Acids Res.*, 2004, **32**, (2), pp. W83–W88
2 Kalaev, M., Bafna, V., Sharan, R.: 'Fast and accurate alignment of multiple protein networks' (Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2008), pp. 246–256
3 Singh, R., Xu, J., Berger, B.: 'Pairwise global alignment of protein interaction networks by matching neighborhood topology' (Research in Computational Molecular Biology, Springer Berlin Heidelberg, 2007), pp. 16–31
4 Liao, C.S., Lu, K., Baym, M., *et al.*: 'IsoRankN: spectral methods for global alignment of multiple protein networks', *Bioinformatics*, 2009, **25**, (12), pp. i253–i258
5 Chindelevitch, L., Liao, C.S., Berger, B.: 'Local optimization for global alignment of protein interaction networks'. Pacific Symp. on Biocomputing, Hawaii, American, January 2010, vol. 15, pp. 123–132
6 Kuchaiev, O., Milenković, T., Memišević, V., Hayes, W., Pržulj, N.: 'Topological network alignment uncovers biological function and phylogeny', *J. Royal Soc. Interface*, 2010, **7**, pp. 1341–1354
7 Milenković, T., Ng, W.L., Hayes, W., Pržulj, N.: 'Optimal network alignment with graphlet degree vectors', *Cancer Inf.*, 2010, **9**, pp. 121–137
8 Kuchaiev, O., Przulj, N.: 'Integrative network alignment reveals large regions of global network similarity in yeast and human', *Bioinformatics*, 2011, **27**, pp. 1390–1396
9 Todor, A., Dobra, A., Kahveci, T.: 'Probabilistic biological network alignment', *IEEE/ACM Trans. Comput. Biol. Bioinf. (TCBB)*, 2013, **10**, (1), pp. 109–121
10 Ibragimov, R., Malek, M., Guo, J., Baumbach, J.: 'GEDEVO: an evolutionary graph edit distance algorithm for biological network alignment', 2013, pp. 68–79
11 Neyshabur, B., Khadem, A., Hashemifar, S., Arab, S.S.: 'NETAL: a new graph-based method for global alignment of protein-protein interaction networks', *Bioinformatics*, 2013, **29**, (13), pp. 1654–1662
12 Tan, J.: 'An adaptive hybrid parallel algorithm for protein-protein interactions network alignment'. Master thesis, Shanghai University, 2013
13 Xie, J., Zhang, S., Wen, T., *et al.*: 'A querying method with feedback mechanism for protein interaction network'. First IEEE Int. Conf. on Healthcare Informatics, Imaging and Systems Biology (HISB), San Jose, CA, July 2011, pp. 351–358
14 Maher, B.: 'ENCODE: the human encyclopaedia', *Natrue*, 2012, **489**, (7414), pp. 46–48
15 Kirk, D.: 'Nvidia: GPU parallel computing architecture'. In IEEE Hot Chips19, IEEE Technical Committee on Microprocessors and Microcomputers (ISMM), Stanford, CA, October 2007, vol. 7, 103–104
16 Manavski, S.A., Valle, G.: 'CUDA compatible GPU cards as efficient hardware accelerators for Smith-Waterman sequence alignment', *BMC Bioinf.*, 2008, **9**, (Suppl 2), pp. S10
17 Sukhwani, B., Herbordt, M.C.: 'GPU acceleration of a production molecular docking code'. Proc. of Second Workshop on General Purpose Processing on Graphics Processing Units, ACM, New York, NY, USA, March 2009, pp. 19–27
18 Vouzis, P.D., Sahinidis, N.V.: 'GPU-BLAST: using graphics processors to accelerate protein sequence alignment', *Bioinformatics*, 2011, **27**, (2), pp. 182–188
19 Yoon, J.S., Jung, W.H.: 'A GPU-accelerated bioinformatics application for large-scale protein interaction networks'. Asia Pacific Bioinformatics Conf. (APBC) poster presentation, Incheon, Korea, January 2011
20 Fossum, E., Friedel, C.C., Rajagopala, S.V., *et al.*: 'Evolutionarily conserved Herpesviral protein interaction networks', *PLoS Pathog*, 2009, **5**, (9), pp. e1000570
21 Altschul, S.F., Gish, W., Miller, W., *et al.*: 'Basic local alignment search tool', *J. Mol. Biol.*, 1990, **215**, (3), pp. 403–410
22 'Human Protein Reference Database', http://www.hprd.org/, 2009 Update