A Novel Framework for Large Scale Metabolic Network Alignments by Compression

Michael Dang
Computer and Information Science and Engineering
University of Florida,
Gainesville, FL 32611
dang@cise.ufl.edu

Ferhat Ay
Computer and Information Science and Engineering
University of Florida,
Gainesville, FL 32611
fay@cise.ufl.edu

Tamer Kahveci
Computer and Information Science and Engineering
University of Florida,
Gainesville, FL 32611
tamer@cise.ufl.edu

ABSTRACT

Although the problem of aligning metabolic networks has been considered in the past, the running time and space complexity of these solutions has so far limited their use to moderately sized networks. In this paper, we address the problem of aligning two metabolic networks, particularly when both of them are too large to be dealt with using existing methods. We develop a generic framework that can be used with any existing method to significantly improve the scale of the networks that can be aligned in practical time. Our framework has three major phases, namely the *compression phase*, the *alignment phase* and the *refinement phase*. For the first phase, we develop an algorithm which transforms the given networks to a compressed domain where they are summarized using fewer nodes, termed *supernodes*, and interactions. In the second phase, we carry out the alignment in the compressed domain using an existing method as our base algorithm. This alignment results in supernode mappings in the compressed domain, each of which are smaller instances of network alignment. In the third phase, we solve each of the instances using the base alignment algorithm to refine the alignment results. Our experiments demonstrate that this method can reduce the sizes of metabolic networks by almost half at each compression level. For the overall framework, we demonstrate how well it increases the performance of an existing alignment method. We observe that we can align twice or more as large networks using the same amount of resources with our framework compared to a recent method for network alignment, namely SubMAP. Our results also suggest that the alignment obtained by only one level of compression captures the original alignment results with very high accuracy.

1. INTRODUCTION

Biological networks encapsulate important information about the roles of different biochemical entities and the interactions between them. Depending on the types of entities and interactions, these networks are segregated into different types, where each network type encompasses a particular set of biological processes. Protein-protein interaction (PPI) networks comprise binding relationships between two or more proteins to carry out specific cellular functions such as signal transduction. Regulatory networks consist of interactions between genes and other DNA segments through their protein or RNA products to control the expression rates of the network elements. Metabolic networks represent sets of chemical reactions that are catalyzed by enzymes to transform a set of metabolites into others to maintain the stability of a cell and to meet its particular needs. Analyzing these networks is essential in order to comprehend the machinery of a cell and to reveal evolutionary differences between different cells and organisms.

An essential type of analysis is the comparative analysis which aims at identifying functionally similar parts shared among different organisms. This is often achieved through alignment of the networks of these organisms. Analogous to sequence alignment which identifies conserved sequences, network alignment reveals connectivity patterns that are conserved among two or more organisms. A number of studies have been done to systematically align different types of biological networks [1–8, 12–17]. For metabolic networks, Pinter *et al.* [20] devised an algorithm that aligns query networks with specific topologies by using a graph theoretic approach. Recently, some of us developed an algorithm that combines both topological features and homological similarity of pairwise molecules to align metabolic networks [2]. We also proposed a method, SubMAP [1, 3], that incorporates subnetwork mappings in metabolic network alignment. A similar method, IsoRank [21], has been applied to find the alignments of PPI networks. IsoRankN [17] extended this algorithm to work for multiple networks and to allow many-to-many mappings of proteins.

Comparative analysis is important particulary for large metabolic networks. Identification of the conserved patterns among metabolic networks across species provide insights for metabolic reconstruction of newly sequenced genome [9], orthology detection [21], drug target identification [22] and identification of enzyme clusters and missing enzymes [10, 18]. However, aligning large scale networks is a computationally challenging problem due to the underlying subgraph isomorphism problem that has to be solved to find the alignment that maximizes the similarity between the query networks. All of the methods we mentioned above either restrict the query topologies and/or their sizes. Even under these conditions, the running times and memory utilization of these methods can still be prohibitive for large query networks. For instance, the method of Pinter *et al.* [20] took around one minute per alignment on a dataset with only small size networks ranging from 2 to 41 nodes. Our earlier method, SubMAP has no limitations on the query topologies and allows mappings of node sets that are connected (i.e., subnetworks). However, allowing subnetworks comes at a cost of increas-

ACM-BCB '11, August 1-3, Chicago, IL, USA

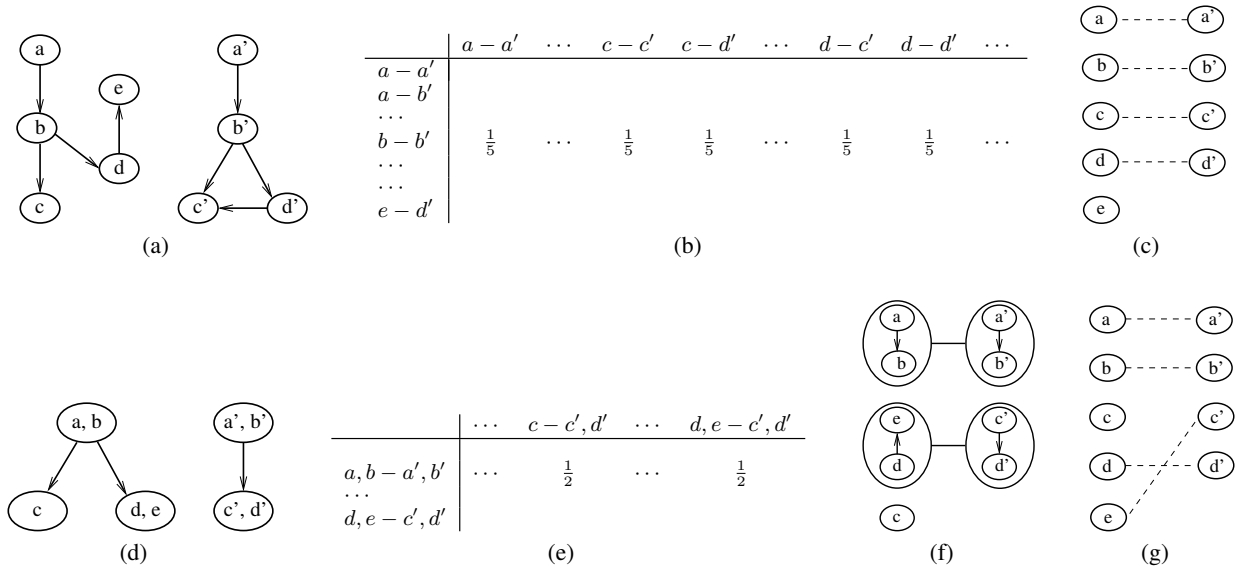

Figure 1: ***Aligning two metabolic networks with and without compression. Top figures (a-c) illustrate the steps of alignment without compression. Bottom figures (d-g) demonstrate different phases of alignment with compression using our framework.*** **(a) Two hypothetical metabolic networks with 5 and 4 reactions respectively. Directed edges represent the neighborhood relations between the reactions. (b) Support matrix of size 20×20 needed for the alignment if compression is not used. We only show the non-zero entries of a single row that corresponds to topological support given by $b-b'$ mapping to possible mappings of its backward and forward neighbors. Five such mappings supported equally are denoted by $\frac{1}{5}$s in the matrix, namely $a-a'$ mapping for the backward neighbors and $c-c'$, $c-d'$, $d-c'$ and $d-d'$ mappings for the forward neighbors. (c) The resulting reaction mappings of alignment without compression. (d) Query networks shown in (a) in compressed domain after one level of compression. (e) Support matrix of size 6×6 needed for the alignment with compression. We only show the entries for the mappings supported by the $a, b-a', b'$ mapping. (f) The resulting mappings from the alignment in compressed domain. (g) The resulting reaction mappings after refinement phase of our framework.**

ing running time that is inherent due to the fact that the number of all connected subnetworks up to a given size can be exponential in the size of the network. For a network of size 70 and subnetwork sizes up to 3, SubMAP takes around 2 minutes and 200 MBs of memory on the average per alignment with a database of 50 networks with sizes ranging from 2 to 57. Therefore, improving the running time and memory utilization of these methods is necessary to leverage the alignment of larger scale networks especially when subnetwork mappings are allowed.

In this paper, we develop a framework that significantly improves the scale of the networks that can be aligned using existing algorithms. Our framework has three major phases, namely the *compression phase*, the *alignment phase* and the *refinement phase*. For the first phase, we develop a compression method that reduces the size of the input metabolic networks by a desired rate. In other words, we transform the query networks from their original domains (see Figure 1(a)) to a *compressed domain* (see Figure 1(d)). A single node in compressed domain corresponds to a set of connected nodes and the edges between them in the original domain. We call each node in the compressed network a *supernode*. For instance, Figure 1(d) depicts the compressed networks of the two input networks in Figure 1(a) when each supernode is allowed to contain up to two nodes (i.e., only one level of compression is allowed). In the second phase, we carry out the alignment in the compressed domain by using an existing network alignment algorithm. Here we use SubMAP as our base alignment method. It is worth noting that, our framework can be used with other alignment methods as well since the performance gain is an inherent property of compression for any base alignment algorithm as long as the query networks can be compressed. Once the compressed networks are aligned, we next consider each mapping of supernodes found by the first phase individually. Each such mapping suggests a smaller instance of network alignment. Figure 1(f) demonstrates this where two such instances exist. For each of these mappings, we solve the alignment problem using the base algorithm. At the end of this refinement phase, the final alignments of reactions are extracted (see Figure 1(g)).

We can best describe the need for our framework on an example. Figure 1 illustrates the difference between aligning two metabolic networks in compressed domain versus aligning them in the original domain without compression. If we use a base alignment algorithm such as SubMAP or IsoRank, the time and space complexity of the algorithm is determined by the size of a data structure, named *support matrix*. Conceptually, this data structure governs the topological similarities between every pair of reaction tuples. Each reaction tuple contains one reaction from each of the two query metabolic networks. A detailed description of this matrix can be found in previous articles describing the IsoRank [21] and SubMAP methods [3]. The size of this support matrix is quadratic in terms of both n and m (i.e., $O(n^2m^2)$) for IsoRank and for SubMAP when only subnetworks of size one are allowed. Figures 1(b) and 1(e) illustrate the support matrices required for alignment starting from the networks shown in Figure 1(a) and 1(d) respectively. As a result of compression by only one level, the size of the matrix we need to create, drops to 6×6 from 20×20 which translates into more than a factor of 10 improvement in the performance of the base method.

Notice that when we compress the network more (i.e., increase the number of compression levels), the compressed network gets smaller in terms of the number of nodes. As a result, we can align the compressed networks faster. However, this comes at the price

of two drawbacks both due to the fact that each supernode tends to contain many nodes from the original domain. First, once we find a mapping for the supernodes in the compressed domain, we still need to align the nodes of each supernode pair. For example, after mapping the supernodes (a, b) and (a', b') shown in Figure 1(f), we need to align the two subnetworks induces by these two supernodes. Thus as the size of the supernodes grow (i.e., as we compress for more levels), the size of the smaller problem instances grow as well and resource utilization bottleneck shifts from the alignment phase to refinement phase. Second, when we use compression the resulting alignment may not be the same as the one found by the original algorithm. For example, one out of four mappings in Figure 1(g) (i.e., $e - c'$) is different than the results of the base algorithm shown in Figure 1(c) (i.e., $e - e'$). We calculate the accuracy as the correlation of the scores calculated for each possible mapping found by our framework in the compressed domain with the scores for these mapping in the original domain found by the base method. Bigger compression rates generally mean less similarity between the results of the two methods (i.e., less accuracy).

Several key questions follow from these observations.

1. How far is our compression method from an optimal compression that produces the compressed network with the minimum number of nodes?
2. What is the right amount of compression? That is, when does compression minimize the running time of our overall framework?
3. How does compression affect the alignment accuracy with respect to the base network alignment method?

In the rest of the paper we address each of these questions in detail.

Our experiments on metabolic networks extracted from KEGG pathway database [19] demonstrate that our compression method reduces the number of vertices and edges by almost half at each level of compression (Section 4.1). As a result of this reduction, we observe significant amount of improvement in running time and memory utilization of our earlier alignment algorithm SubMAP (Section 4.2). Lastly, we analyze the accuracy of our framework as compared to the base alignment algorithm. The results suggest that the alignment obtained by only one level of compression captures the original alignment results with very high accuracy and the accuracy decreases with further levels of compression (Section 4.3).

In summary, our technical contributions are:

- We devise an efficient framework for the network alignment problem that employs a scalable compression method which shrinks the given networks while respecting their topology.
- We prove the optimality of our compression method under certain conditions and provide a bound on how much our compression results can deviate from the optimal solution in the worst case.
- We provide a mathematical formulation that serves as a guideline to select an optimal number of compression levels depending on the input characteristics of the alignment.

The organization of the rest of this paper is as follows. Section 2 presents the method we propose for compressing the networks. Section 3 describes the remaining phases of our framework that performs the alignment in compressed domain and analyzes its complexity. We report our experimental results on a set of metabolic networks in Section 4. Section 5 briefly concludes the paper.

$P = (V, E)$, $\bar{P} = (\bar{V}, \bar{E})$	Query metabolic networks
$V, \bar{V}$	Sets of all reactions of the query networks
$r_i \in V, \bar{r_j} \in \bar{V}$	Reactions of the query networks
$n = \|V\|, m = \|\bar{V}\|$	Sizes of the query networks
c, 2^c	Compression level and compression rate
$P^c = (V^c, E^c)$	P after c levels of compression
$C_i = (\bar{V}_i, \bar{E}_i)$	A connected component of network P
$N(v_a)$, $deg(v_a)$	The set of neighbors and degree of node v_a
$\|v_a\|$	Number of reactions that are contained in v_a
v_{ab}	A supernode containing the nodes v_a and v_b
k	Parameter for the largest subnetwork size
$\mathcal{R}_k, \bar{\mathcal{R}}_k$	Sets of all subnetworks of size at most k
$R_i, \bar{R}_j$	Subnetworks of the query networks
N_k, M_k	Numbers of all subnetworks of size at most k

Table 1: ***Commonly used symbols in this paper.***

2. COMPRESSION PHASE

In this section, we describe the method we develop to compress the query networks. Before going into detail, it is important to state that we are using a reaction-based model for representing metabolic networks throughout this paper. Formally, we represent a metabolic network with $P = (V, E)$ where V is the set of all reactions of the network and E is the set of directed edges between them. An edge $e_{ij} \in E$ exists if and only if the reaction v_i has at least one output compound which is an input for the reaction v_j. In the following, we first describe our compression method (Section 2.1). We use the shorthand notation MDS (minimum degree selection) to refer to this method in the rest of the paper. We, then, prove the optimality of MDS under certain conditions and provide an upper bound for the number of compressions that can be missed by this method with respect to the optimal compression (Section 2.2).

2.1 Minimum Degree Selection (MDS) Method

Let $P = (V, E)$ be the reaction-based representation of a metabolic network and c denote the user specified parameter for the desired level of compression. For $x = 1, ..., c$, we denote the compressed form of P after x compression levels with $P^x = (V^x, E^x)$. To simplify our notation, we assume that $P^0 = P$. We construct P^x from P^{x-1} for each $x = 1, ..., c$. Each $v \in V^x$ is either a node from V^{x-1} or a supernode that contains two nodes of V^{x-1}. In summary, we construct V^x from V^{x-1} in a number of consecutive steps. At each step, we choose a pair of connected nodes in V^{x-1} that are not compressed in earlier steps of the current compression level. We then merge this node pair into a supernode and add it to V^x. We repeat these steps until there is no such node pair in V^{x-1}. Assume that the number of such steps is t for compression level x. We denote the state of the network after the ith step during the xth level of compression as $P_i^x = (V_i^x, E_i^x)$ (Figure 2(b)). Note that, $V_t^x = V^x$ and $V_i^x \subseteq V^{x-1} \cup V^x$ for each $i = 1, ..., t$ as the nodes of V_i^x are either singleton nodes from V^{x-1} or supernodes from V^x.

We are now ready to discuss how we compress P^{x-1} to get P^x. We define the *degree* of a non-compressed node v in a given network as $deg(v) = indeg(v) + outdeg(v)$, where $indeg(v)$ ($outdeg(v)$) denotes the number of incoming edges from (out-going edges to) non-compressed nodes in the network. We say that two nodes in a network are neighbors if they are connected by at least one edge. We denote the set of neighbors of a node v with $N(v)$. We start the compression by initializing $V_0^x = V^{x-1}$, $E_0^x = E^{x-1}$. Then, while there exists a non-compressed node with degree greater than

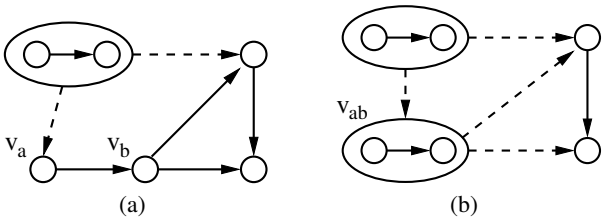

Figure 2: ***One compression step of the MDS method on a hypothetical metabolic network P.*** **Small circles represent reactions and big circles represent supernodes that result from earlier steps of compression. A solid arrow represents an edge between two non-compressed nodes in the current compression level. A dashed arrow denotes an edge between a supernode and another node in the network. While calculating the degrees of the non-compressed nodes, only the solid arrows are taken into account. (a) The state of network P during compression level x before the ith intermediate step (i.e., P^x_{i-1}). The node with the minimum degree is denoted with v_a and its first neighbor is denoted with v_b. (b) The state of this network after the ith compression step (i.e., P^x_i). We denote the node resulted from the compression at this step with v_{ab}.**

zero at the current state of the network, say P^x_{i-1}, we apply the next step, the ith step, of compression to obtain P^x_i from P^x_{i-1}. Figure 2 depicts the states of an example network before (Fig. 2(a)) and after (Fig. 2(b)) the ith step of compression. We start the ith step by selecting a node with minimum positive degree among the nodes in V^x_{i-1}. If there are more than one such node, we select the first one among them. In our example in Figure 2(a), the node with minimum degree is unique and is shown by v_a. We use the term minimum degree as a shorthand for minimum positive degree to exclude singleton nodes. This way we ensure that $deg(v_a) > 0$ and $N(v_a)$ is non-empty. We select one such neighbor from $N(v_a)$, say v_b. The only node in $N(v_a)$ in Figure 2(a) is denoted with v_b. We, then, merge v_a with v_b to form the supernode $v_{ab} = \{v_a, v_b\}$. Figure 2(b) illustrates this newly created node v_{ab}. This is the only compression to be done at the ith compression step. Next, we create the new node set as $V^x_i = V^x_{i-1} \cup \{v_{ab}\} - \{v_a, v_b\}$. For creating the edge set E^x_i, we initialize it to E^x_{i-1} and remove all the incoming and out-going edges of v_a and v_b from it. Then, we insert an incoming edge to v_{ab} from each node in $V^x_{i-1} - \{v_a, v_b\}$, which has an out-going edge to either v_a or v_b in the previous edge set E^x_{i-1}. We insert out-going edges from v_{ab} to other nodes in a similar manner. Figure 2 illustrates the changes in the edge set after creating v_{ab}. Notice that for each $i = 1, ..., t$, the set V^x_i contains a mixture of nodes and supernodes. After each such step, the size of the network decreases by one and the number of edges of the new network decreases at least by one. For instance in Figure 2, the number of nodes dropped from five to four and the number of edges dropped from six to five. The compression of P^{x-1} to get P^x continues by applying another compression step until there are no more non-compressed nodes with positive degree.

The discussion above describes the intermediate compression steps of the MDS method to perform a single level of compression on a given network. Given a compression level c, for each level $x = 1, ..., c$, we apply the same compression steps on $P^{x-1} = (V^{x-1}, E^{x-1})$ by initially treating P^{x-1} as a non-compressed network with no supernodes. As a result of this process, after finishing the xth level of compression, the actual number of reactions that each node of V^x can contain is assure to be in the interval $[1, 2^x]$. The limitation on the number of reactions in each node allows the MDS method to respect and highly preserve the initial topology of the query networks. This is very important for the alignment as it makes significant use of the network topologies. Additionally, the bound on the number of reactions in each supernode translates to a uniform compression for both networks which limits the sizes of the smaller alignment problems we can encounter in the refinement phase (see Section 3.3). This allows us to keep under control the complexity and the running time of the refinement phase of our alignment framework.

2.2 Optimality Analysis for MDS

In the previous section, we described in detail the compression method (MDS) we use in our framework. Ideally, it is preferable to compress the given network as much as possible at each compression level. This is because smaller network size often implies smaller time and memory usage for the alignment. We say that a compression is *optimal* if the resulting compressed network contains the smallest number of nodes among all possible compressions with the restriction that each non-compressed node can be merged with at most one other non-compressed node at each compression level. We name the hypothetical optimal compression method that can achieve the best possible compression rate as OPT. In the rest of this section, we analyze the optimality of our MDS method under different conditions. We first consider each connected component of the input network that will be compressed separately and then integrate their results to generalize our analysis for networks with arbitrary topologies.

We start by introducing the notation we use in this section to handle networks with more than one connected components. Let P be a metabolic network with r connected components. We denote these components by $C_1 = (\hat{V}_1, \hat{E}_1)$, $C_2 = (\hat{V}_2, \hat{E}_2)$, ..., $C_r = (\hat{V}_r, \hat{E}_r)$ such that $P = (\bigcup_{j=1}^{r} \hat{V}_j, \bigcup_{j=1}^{r} \hat{E}_j)$. Let $C = (\hat{V}, \hat{E})$ be an arbitrary component of P and $*^x$ represent the compressed form of C after x levels of compression using either the MDS method or OPT that achieves the optimal compression. We use $*$ (star) as a generic symbol to avoid introducing new symbols for each compressed component in places where only their sizes are of relevance. We use $MDS(C, *^x)$, $OPT(C, *^x)$ to denote the total number of compression steps performed to transform C into its compressed form after x levels of compression by using the corresponding methods. Recall that each compression step reduces the network size by one. Thus, the bigger these values ($MDS(C, *^x)$ and $OPT(C, *^x)$) the better they are in terms of compression rate. The first and second arguments in this notation can be any state of a connected component or a network at any point during the compression. For instance, $OPT(C^x_i, *^x)$ denotes the number of compression steps taken by OPT starting from $(i+1)$th intermediate step of the xth level until the xth level of compression is completed.

In the following, we first prove that the MDS method makes an optimal choice in terms of which two nodes to compress at each compression step if there exists a node with degree one in the current state for a given component. We, then, show that if no node with degree one exists at a compression step taken by MDS can increase the size of the compressed component by at most one as compared to the one found by OPT. Finally, by aggregating the results from each component, for a given metabolic network P and a compression level c, we develop an upper bound on the size of the compressed networks obtained by MDS with respect to the size of network that can be obtained by the optimal method.

LEMMA 1. *Let $C = (\hat{V}, \hat{E})$ denote a connected component of a given metabolic network P. Let $C^x_i = (\hat{V}^x_i, \hat{E}^x_i)$ denote the state*

of C after the ith step of the xth compression level. If there exists a node in $\hat{V}_i^x$ with degree one, then the compression step taken by the MDS method to create the next state C_{i+1}^x is optimal. Formally,

$$OPT(C_i^x, *^x) = 1 + OPT(C_{i+1}^x, *^x) \quad (1)$$

Proof: Omitted. □

LEMMA 2. *Let $C = (\hat{V}, \hat{E})$ denote a connected component of a given metabolic network P. Let $C_i^x = (\hat{V}_i^x, \hat{E}_i^x)$ denote the state of C after the ith step of the xth compression level. If the node with minimum degree in $\hat{V}_i^x$ has degree greater than one, then the compression step taken by MDS to create the next state C_{i+1}^x can lead to a network that has size at most one larger than the compressed network that is obtained from the state C_i^x by OPT. Formally,*

$$OPT(C_i^x, *^x) \leq 2 + OPT(C_{i+1}^x, *^x) \quad (2)$$

Proof: Omitted. □

Using lemmas 1 and 2, Theorem 1 develops an upper bound on the number of compression that can be missed by MDS with respect to the optimal compression.

THEOREM 1. (OPTIMALITY BOUND FOR MDS) *Let P be a metabolic network with r connected components $C_1 = (\hat{V}_1, \hat{E}_1)$, ..., $C_r = (\hat{V}_r, \hat{E}_r)$ such that $P = \bigcup_{j=1}^{r} C_j$ and c be a positive integer given as the desired number of compression levels. Let $C = (\hat{V}, \hat{E})$ denote an arbitrary connected component of P. Also, let s represent the number of intermediate steps for which no non-compressed nodes with degree one is found during the compression from P to P^c by the MDS method.*

Then, each of the following statements hold:

1. $OPT(C^{x-1}, *^x) \leq 2\ MDS(C^{x-1}, *^x)$ *for* $x = 1, ..., c$.
2. $OPT(P, *^c) \leq s + MDS(P, *^c)$
3. $OPT(P, *^c) \leq min\{\ 2\ MDS(P, *^c),\ s + MDS(P, *^c)\ \}$.

Proof: Omitted. □

Another way of interpreting Theorem 1 is to transform it to an upper bound on the size of the compressed network generated by MDS in terms of the one that can be obtained by OPT. By carrying out this transformation, we answer the first question we pointed out in the introduction which is "How far is our compression method from the optimal compression?". We do this as follows. Let P be a network of size n. Given compression level c, let us represent the number of compressions steps of the OPT method with $\theta = OPT(P, *^c)$. Also, let n_{OPT} and n_{MDS} denote the sizes of the compressed networks obtained by the OPT and MDS methods respectively. By the bound given in Theorem 1, we know that $MDS(P, *^c) >= \lceil \frac{\theta}{2} \rceil$. Therefore, we can write $n_{OPT} = n - \theta$ and $n_{MDS} \leq n - \lceil \frac{\theta}{2} \rceil$. Also, we know by definition that $\theta \leq \sum_{x=1}^{c} \lfloor \frac{n}{2^x} \rfloor$. Using this inequality, we get:

$$n_{OPT} \geq n - \sum_{x=1}^{c} \lfloor \frac{n}{2^x} \rfloor\ ,\ n_{MDS} \leq n - \lceil \sum_{x=1}^{c} \lfloor \frac{n}{2^{x+1}} \rfloor \rceil \quad (3)$$

If we examine the ratio $\frac{n_{MDS}}{n_{OPT}}$, for $c = 1$ we get $\frac{n_{MDS}}{n_{OPT}} \leq \frac{3}{2}$ for arbitrary n (details omitted). This demonstrates that after one level of compression, the size of the compressed network found by our method is at most 1.5 times the size of the optimal network. For $x = 1, 2, ..., c$, this ratio is proportional with $(1.5)^x$. We can also use the bound on number of compression steps given in the second statement of Theorem 1 to gather a similar upper bound on the size of the compressed network found by MDS. The tighter of these two upper bounds on the network size can be calculated during the execution of the MDS method and reported as an indicator of how much room is left for improving the compression.

3. ALIGNMENT FRAMEWORK

We described the first phase, namely the compression phase in detail in Section 2. Here, we first summarize the base alignment method, SubMAP [3], we use in our framework in Section 3.1. Then, we explain the two remaining phases of our framework, namely the alignment phase and the refinement phase. The alignment phase follows the compression phase and utilizes the base method to find an alignment in compressed domain (Section 3.2). The refinement phase applies the base method on the mappings found in previous phase to further refine the alignment results (Section 3.3). After describing all the phases, we analyze the complexity of each phase and combine them to obtain the complexity of the entire framework (Section 3.4). Last, we provide a guideline for selecting the compression level that is expected to give the best performance gain reached by our framework with respect to the base alignment method (Section 3.5).

3.1 Overview of SubMAP

Here, we take a small detour and explain SubMAP, a recent method for aligning metabolic networks when they are not compressed. We pick SubMAP method for its high accuracy and biological relevance as it considers subnetworks of the given networks during the alignment. A *subnetwork* of a network is a subset of the reactions of that network such that the induced undirected graph of this subset is connected. Given two metabolic networks $P = (V, E)$ and $\bar{P} = (\bar{V}, \bar{E})$ and a positive integer k, SubMAP aims to find a set of mappings between the reactions of P and $\bar{P}$ with the largest similarity score, such that: (i) Each reaction in P ($\bar{P}$) can map to a subnetwork of $\bar{P}$ (P) with at most k reactions (ii) Each reaction of P and $\bar{P}$ can appear in at most one mapping.

The first step of SubMAP is to create the set of all possible subnetworks of size at most k for each query network. We denote the number of these subnetworks for P and $\bar{P}$ with N_k and M_k respectively. The second step of SubMAP is to calculate pairwise similarities between each pair of these subnetworks one from P and one from $\bar{P}$. We refer the reader to Ay *et al.* [3] for the details of the pairwise similarity score. The step that dominates the time and space complexity of SubMAP is the third step. The aim of this step is to create a similarity score that combines pairwise similarities with the topological similarity of the networks. A data structure named the *support matrix* is created for this purpose. The size of this matrix is quadratic in terms of the number of subnetworks of both query networks. In other words, the support matrix requires $O({N_k}^2 {M_k}^2)$ space. This complexity is very important as it is the dominating factor in the overall time and space complexity of SubMAP. The next two steps of the algorithm are to combine topological similarity with pairwise node similarities and to extract the alignment as a set of subnetwork mappings of P and $\bar{P}$.

3.2 Alignment Phase

The SubMAP method described above aligns the networks $P = (V, E)$ and $\bar{P} = (\bar{V}, \bar{E})$ in their original form. Our framework

first compresses each of these networks to reduce their sizes and then aligns the compressed networks instead of P and $\bar{P}$. In this section, we explain how we align the compressed networks P^c and $\bar{P}^c$ that are in the compressed domain of level c using SubMAP with a given parameter k.

Let us first consider $P^c = (V^c, E^c)$. Each node v_a in V^c is a supernode of the reactions in V. Also, by the working of our compression method, we know that each supernode v_a contains at most 2^c reactions. An edge from the node v_a to the node v_b exists in E^c if and only if at least one reaction in v_a has an edge to one reaction in v_b in E. The same arguments hold for the other network $\bar{P}^c$ as well. To align these compressed networks, we consider their nodes, which are supernodes of reactions, as if they are the reactions of the metabolic networks P^c and $\bar{P}^c$. This way, we can directly apply SubMAP to align these networks. As far as the operation of the SubMAP method is concerned, this is no different than aligning two networks that are identical to these networks but are in the original domain. The difference is in the interpretation of the intermediate steps and the form of the mappings found by the alignment. For instance, for the first step of SubMAP, we enumerate the reaction subnetworks of size at most k in the original domain, whereas in the compressed domain we enumerate the subnetworks of supernodes where each supernode can contain more than one reaction and the number of such supernodes in one subnetwork is at most k. Similarly, we calculate the pairwise similarity, the support matrix and the conflict graph for the subnetworks of supernodes (i.e., nodes of V^c) instead of subnetworks of reactions (i.e., nodes of V). The resulting alignment gives us a set of mappings between the subnetworks of P^c and $\bar{P}^c$. We can think of these mappings as a high level view of the alignment between the networks P and $\bar{P}$. For instance, from Figure 1(f) one can immediately see that the resulting alignment will map node a either to node a' or node b' and that these are the only options for node a which is imposed by the higher level supernode mapping $(a, b - a'b')$. In the next phase, we consider each of these supernode mappings as smaller instances of the alignment problem and solve them to obtain a more refined alignment of P and $\bar{P}$.

3.3 Refinement Phase

Each mapping found by the alignment phase is a subnetwork pair where one is from P^c and the other is from $\bar{P}^c$. The mappings found by SubMAP can have up to k nodes in one subnetwork and only one node in the other. If we denote a subnetwork of P^c with R_i^c and a subnetwork of $\bar{P}^c$ with $\bar{R}_j^c$, the resulting mappings of the alignment phase will be in the form $(R_i^c, \bar{R}_j^c)$. We can assume, without loss of generality, for this specific pair that R_i^c contains up to k nodes of P^c and $\bar{R}_j^c$ contains a single node of $\bar{P}^c$. Each node contained in either of these subnetworks is a supernode that contains either one node or two nodes and an edge between them in the previous level of compression, namely the $(c-1)$th level. For both R_i^c and $\bar{R}_j^c$, we decompress their nodes by one level by retrieving the connectivity between these nodes in the $(c-1)$th compression level that was encapsulated in the cth level. This decompression results in at most $2k$ nodes from $(c-1)$th level for R_i^c and at most 2 nodes from $(c-1)$th level for $\bar{R}_j^c$. We then recursively align these smaller networks generated from R_i^c and $\bar{R}_j^c$ by using SubMAP until the original domain (i.e., $c = 0$) is reached. At the $(c-x)$th recursive step, the sizes of two networks to be aligned can be at most $k\ 2^x$ for one network and 2^x for the other.

Figure 1(f) illustrates this on a concrete example. The network on the left has two supernodes (i.e., (a, b) and (e, d)) each containing two nodes with an edge between them and one supernode (i.e., (c)) which contains only one node from the previous level of compression. The one on the right has two supernodes with two nodes in each. To understand how decompression by one level works, we can focus on the supernode mapping (e, d) - (c', d') which is found in compression level one. We can think of decompression as removing the circles that surround these supernodes to get back the connectivity within their nodes in the previous compression level. In our case, this leads to the small networks $d \rightarrow e$ and $c' \rightarrow d'$. We align these small networks recursively using SubMAP and report their final alignment in only one recursive call since the compression level is only one for this case. Also, since $k = 1$ is used for the ease of this example, the sizes of the networks, in terms of the nodes in original domain, on each side are at most 2 for the recursive call from $c = 1$ as can be seen from Figure 1(f) (i.e., $k\ 2^c = 2^c = 2$ for $k = c = 1$).

3.4 Complexity Analysis

Having finished the discussion of all the three phases, now we can analyze the overall complexity of our framework. We start from the first phase which is compression of the input networks P and $\bar{P}$ by c levels. We first calculate the complexity of the first compression level for the network P with size n. At each compression step, MDS first searches for a minimum degree node. Once it finds this node, it picks one of its neighbor nodes and merges these two nodes. After this merging, it updates the degrees of all the neighbors of each of the merged nodes. The first two of these operations take $O(log\ n)$ time if proper data structures are used and the last one can take $O(n)$ in the worst case. Since the size of network P is n, there can be at most $\lfloor \frac{n}{2} \rfloor$ compression steps during the first level of compression. Hence, the complexity of the compression for the first level is $O(n^2)$. Since the input sizes of this level is larger than all the next levels, we can safely assume that each of these next levels also take $O(n^2)$ and the complexity of compression by c levels is therefore $O(cn^2)$. Even though this is not a tight bound, it is sufficient at this point for the complexity of the next two phases will dominate it. Since we compress both networks, the overall complexity for the compression phase is:

$$O(c(n^2 + m^2)). \tag{4}$$

For the analysis of the next phases, we make two assumptions both of which are supported by experimental evidence on the topological properties of metabolic networks. Our first assumption is that at each level of compression our method reduces the network size by half. In other words, if the sizes of our query networks are n and m, then the sizes of the compressed networks after c levels by the MDS method are $n_{MDS} = \lceil \frac{n}{2^c} \rceil$ and $m_{MDS} = \lceil \frac{m}{2^c} \rceil$ respectively. This is mainly because metabolic networks contain many nodes with low degrees [11]. Our experiments on a large dataset of networks summarized in Table 2 supports this as well. The second assumption is that the number of subnetworks is a constant multiple of the network size for small k values. In other words, $N_{MDS} = \alpha(k)\ n$ and $M_{MDS} = \beta(k)\ m$ where $\alpha(k)$ and $\beta(k)$ are functions of k but are independent of n and m respectively. Our earlier analysis in Ay *et al.* [3] demonstrated that the number of subnetworks for $k = 3$, which is the largest k value we use here, is in the order of $5|V|$ for a large set of metabolic networks.

We are now ready to analyze the complexity of the second phase which is the alignment phase. By the first assumption, we know that the sizes of P^c and $\bar{P}^c$ are $n_{MDS} = \lceil \frac{n}{2^c} \rceil$ and $m_{MDS} = \lceil \frac{m}{2^c} \rceil$

respectively. By the second, we have the number of subnetworks of these networks as $N_{MDS} = \alpha(k)\ n$ and $M_{MDS} = \beta(k)\ m$ for a given k. Also, we know that the complexity of SubMAP is quadratic in terms of N_{MDS} and M_{MDS} from Section 3.1. Therefore, the complexity of the second phase is:

$$O(\frac{\alpha(k)^2\beta(k)^2n^2m^2}{2^{4c}}). \quad (5)$$

The complexity of the refinement phase has two factors in it. The first one is the number of mappings found by the alignment phase. Since we know that SubMAP allows each node of both networks to be reported in at most one mapping, we have a trivial upper bound on the number of possible mappings in terms of n and m. The biggest number of mappings is reported when all the subnetworks of both networks are singletons. In this case, the number of reported mappings is the minimum of n and m. We can assume without loss of generality that $n < m$ and hence this number is $O(n)$. The second factor is the sizes of each of these $O(n)$ smaller alignment problems that needs to be solved by SubMAP again to refine the mapping results. As we discussed in Section 3.3, the sizes of the networks created by decompressing the mapped subnetworks by one level are at most $k\ 2^c$ on one side and at most 2^c on the other. The number of subnetworks that can be created from these networks are $\alpha(k)\ k\ 2^c$ and $\beta(k)\ 2^c$ for the corresponding sides. Therefore, each mapping can be refined by decompressing and applying SubMAP which is $O(\alpha(k)^2\ k^2\ 2^{2c}\ \beta(k)^2\ 2^{2c})$. We do this refinement for $O(n)$ times in the worst case, hence the complexity of the refinement phase is:

$$O(\alpha(k)^2\beta(k)^2nk^2 2^{4c}). \quad (6)$$

Combining the results of Equations 4, 5 and 6, we can see that the overall complexity of our method is determined by the second or the third phase depending on the value of c. For small values of c and k such as 1, 2 and 3, the second phase dominates the overall complexity. Larger values of c results in a costlier refinement phase and a less expensive alignment phase. Very large values of k imply exponentially many subnetworks in which case the above complexity analysis would not hold and the alignment problem may become intractable with or without compression.

3.5 How Much Should We Compress?

In this section, we provide a guideline for selecting a value for compression level c that results in the minimum running time, among other possible values, for our framework to align the query networks with for a given k. We make extensive use of the complexity results found in Section 3.4 in the proof of the below theorem which formulates the optimal c for a given k value and the two query networks with sizes n and m. This theorem answers the question "What is the right amount of compression that we need to use in order to minimize the running time of our framework?".

THEOREM 2. (OPTIMAL LEVEL OF COMPRESSION) *Let $P = (V, E)$, $\bar{P} = (\bar{V}, \bar{E})$ be two metabolic networks with sizes n and m respectively, and k be a given positive integer. Assume without loss of generality that $n < m$. Then, the compression level c that gives the optimal compression is:*

$$c = \frac{\log_2(nm^2k^{-2})}{8}. \quad (7)$$

Proof: Omitted. □

The value obtained from the above discussion is not necessarily a real number. We suggest using the nearest integer to this value as the number of compression levels in our alignment. Next, we want to give a few examples for to see what Theorem 2 implies in practice. Assume we have two networks with sizes $n = 20$, $m = 20$ and we want to align them using our framework for $k = 2$. Plugging these number in Equation 7, we get:

$$c = \frac{\log_2(2000)}{8} = \frac{\log_2(10.966)}{8} \cong 1.37$$

If we round this to the nearest integer, the Equation 7 suggests that we use only one level of compression for this alignment problem. We can carry the calculations similarly for another set of inputs $n = m = 80$ and $k = 2$ which gives around 2.15, suggesting 2 levels of compression is likely to provide the best running time improvement for this instance.

4. EXPERIMENTAL RESULTS

In this section, we experimentally evaluate the performance of our framework. First, we measure the compression rates achieved for different values of c. We, also, compare the compression rates of the MDS method that selects the first node with minimum degree at each step with the rates obtained from a number of different compressions obtained by randomizing this node selection (Section 4.1). Then, we analyze the gain in running time and memory utilization achieved by our framework for different values of c and k (Section 4.2). Last, we examine the accuracy of the alignments found by our framework. We measure the accuracy in terms of the Pearson's correlation coefficient between the scores of mappings resulted from alignment in compressed domain and the ones resulted without compressing the networks (Section 4.3).

Dataset: We use the metabolic networks from the KEGG pathway database [19]. We downloaded all metabolic networks with at least 10 reactions for 20 different organisms. This resulted in 620 networks in total with sizes ranging from 10 to 97. In order to obtain larger networks, we combined all the metabolic networks belonging to carbohydrate metabolism for each organism into one larger network for 10 of these organisms. Similarly, we combined the networks of cofactor and vitamins metabolism of each of these 10 organisms. The sizes of these 20 combined networks range from 59 to 279 reactions. In total, our dataset contains 640 metabolic networks that have sizes in the interval [10, 279].

Implementation and system details: We implemented our compression and alignment algorithms in C++. We ran all the experiments on a desktop computer running Ubuntu 10.10 with 4 GB of RAM and two 2.66 GHz processors.

4.1 Evaluation of Compression Rates

The efficiency of our alignment framework depends on how much the query metabolic networks can be compressed. For this reason, in this experiment, we measure the number of nodes and edges of the networks in our dataset before and after compressing them. Recall that the MDS method selects the first node from the list of nodes with minimum degree at each intermediate step and compresses it with its first neighbor from the list of its neighbors. In order to evaluate stability of our compression method, for each network in our dataset we generated a number of different compressed networks by randomizing the minimum degree selection step of our method. In the following, we examine how much compression we achieve by the MDS method and discuss its stability.

Table 2 summarizes the compression rates achieved by our method

Table 2: ***Summary of compression rates for all the networks in our dataset.*** **We create six intervals according to number of reactions in these networks. Each row, corresponding to one such interval, shows the average number of nodes and edges before compression (i.e., c=0) and after compression of different levels (i.e., c = 1, 2, 3) both by the MDS method (top entries with no cell color) and by its randomized version averaged over 10 different runs (bottom entries with gray cell color).**

Network size intervals	*Average Number of Nodes*				*Average Number of Edges*			
	c=0	**c=1**	**c=2**	**c=3**	**c=0**	**c=1**	**c=2**	**c=3**
[10, 20)	14.05	8.85	6.37	5.25	18.37	9.22	4.80	1.95
		8.86	6.43	5.17		9.21	4.85	2.18
[20,40)	26.74	15.91	10.55	8.10	46.32	25.83	15.44	6.82
		15.91	10.72	8.05		25.83	15.66	7.72
[40,60)	47.95	30.81	21.64	17.67	76.76	45.07	32.00	20.24
		30.80	21.76	16.99		44.91	33.18	21.52
[60,80)	69.90	40.10	26.00	19.50	198.30	113.70	74.40	45.90
		40.16	26.15	18.89		113.96	75.84	48.41
[80,100)	88.25	47.75	27.75	19.88	309.00	165.63	98.13	59.63
		47.79	27.69	19.36		165.83	99.38	56.83
[100,100+)	173.44	98.44	61.31	47.44	1619.88	930.81	515.44	248.19
		98.32	61.33	45.51		924.46	518.18	276.47
All	26.66	16.06	10.83	8.56	78.82	44.22	25.46	12.48
		16.07	10.94	8.39		44.05	25.75	13.72

for networks of different sizes. We divide all the metabolic networks in our dataset into six groups according to the number of their reactions (i.e., network size). The first column in Table 2 lists the network size intervals we used for each group. Each row of this table shows the number of nodes and edges averaged over all the networks in this group before and after compression. The two columns with $c = 0$ correspond to the average number of nodes and edges of the networks with no compression respectively. For $c \in \{1, 2, 3\}$, we split each row corresponding to an interval into two. The upper part denotes the average node and edge numbers for the compressed network if the MDS method is used. The lower part with gray cell color represents the numbers gathered when we introduce randomization in the node selection. That is at for compression step at which there are more than one nodes with minimum degree, we select one node among them randomly and we repeat this process 10 times to obtain different compressions. Each value with gray cell color in Table 2 denotes the average of the corresponding value over these 10 different runs of compression.

One conclusion that can be drawn from Table 2 is that independent of the network size, our compression method performs well in practice. On the average, with only one level of compression we achieve network sizes that are 62%, 68% and 77% of the network size in the previous compression level for $c = 1$, 2 and 3. In other words, our method compresses the entire dataset down to 62%, 42% and 33% of the sizes of original networks for $c = 1$, 2 and 3 respectively. These compression values suggest that our framework has great potential in scaling the network alignment to large metabolic networks. As an example, consider the row corresponding to interval [80, 100) in Table 2. We see that instead of aligning a network with 88 nodes and 309 edges, we can apply three levels of compression first and do the alignment with a significantly smaller network with only 20 nodes and 60 edges. Another observation is that, we get the most of the reduction in network size after the first compression level. That is, our method compresses the networks aggressively for $c = 1$ and achieves 62% compression rate which is close to the half of the size of the networks. As we increase the value of c, the actual rate of compression at one level reduces.

Another result of this experimental setup is that the MDS method is stable. In other words, it is not affected by the choice of the node to compress as long as that node is selected from among the nodes with minimum degree. Focusing on the row corresponding to interval [80, 100), we can observe that all of the differences for $c = 1$, 2, 3 are less than one for both the number of nodes and the number of edges. Some other rows have slightly bigger differences (e.g., the row corresponding to interval $[100, 100+]$), however, none of them are significant. *From the results of this experiment, we conclude that our compression method is stable and it serves as an efficient first phase of our alignment framework since it achieves good compression rates on a large dataset of metabolic networks.*

4.2 Evaluation of Running Time and Memory Utilization

In order to understand the capabilities and limitations of our framework, we examine its performance in terms of its running time and memory utilization on a set of networks from our dataset that is extracted from KEGG. We used different combinations of k and c values as well as different networks with sizes from a broad spectrum. When the value of c is equal to zero, the alignment is carried out completely by a single application of SubMAP without any compression. This provides us a mechanism to measure how much performance gain is achieved by our compression based framework with respect to SubMAP that uses no compression.

To measure the change in resource utilization for different values of k, c and the network sizes, we generated a large number of alignments between the networks in our dataset. We create a query set by selecting networks of varying sizes from the interval $[20, 279]$. We also select 10 networks with sizes very close to multiples of ten starting from 10 to 100 as our database set. The average network size of this set is 55, which is greater than double the average size for the overall dataset (26.66). For each query network we run an alignment with all the networks in the database set for each k value, c value combination. We have twelve combinations in total for $k =$ 1, 2, 3 and $c = 0$, 1, 2, 3.

Figure 3(a) illustrates the average running time of our framework for query networks with increasing number of reactions when $k =$ 1 is used for each alignment. We plot all the results for all four different compression values and also draw the fitting curves to better illustrate the trend in the increase of running time. We can observe from the figure that each additional compression level improves the running time over the previous one for all query sizes when $k = 1$. We obtain the largest gain in running time by only one level of compression for the first level. This is expected considering that the first level of compression achieved the largest compression rate as shown in Table 2. The second compression level improves the running time by a smaller factor compared to the first and by a larger factor compared to the third level. For $k = 1$ we were able to plot all the points for all c values as the running time for even the largest query network (i.e., size 279) with no-compression (i.e., c $= 0$) is still practical, around 100 seconds.

Figure 3(b) is created similar to Figure 3(a) but $k = 2$ is used instead. We observed that most of the alignments in the original domain $c = 0$ did not finish in less than a cutoff time which we set as one hour. This is because the number of subnetworks increased significantly when the value of k is increased to two. Therefore, we did not plot the running time values for $c = 0$. To simplify the figure, we also omitted the plot for $c = 1$ as it is very similar to the results for $c = 2$. We make the decision of whether or not to plot a point by looking at the percentage of alignment queries that completed before the cutoff time. If out of ten possible align-

ments for a specific query network, at least eight (i.e., 80%) are completed before the cutoff, then we plot the running time for this query size and for the corresponding c value in our figure. Focusing on a specific network size in Figure 3(b), we can observe significant decrease in running time due to compression by three levels instead of two. For instance if we look at the network size that is closest to 200, the average running time of a query is around 20 minutes for $c = 2$ and it drops to almost 2 minutes when $c = 3$. Additionally, the points that are omitted from the Figure 3(b) due to the one hour cutoff time suggest that by using the correct amount of compression, our framework makes it possible to align networks that could not be aligned with the base method which is SubMAP in our case. We believe this is an important step in leveraging larger scale network alignments for they provide a more complete picture of functional similarities and evolutionary differences between the metabolic networks of two or more organisms.

We omit the details and figures for the memory utilization due to the space constraints. We mention briefly that on the average the memory required for alignments in compressed domain is around 30% of that needed for alignment with no compression using the SubMAP method. *Therefore, our framework demonstrates a great potential in overall to provide significant improvement in both the running time and the memory utilization of the base alignment method. This allows us to align large networks that could not be aligned by existing methods by utilizing the same hardware.*

4.3 Accuracy of the Alignment Results

We conclude our experimental results by answering the last question that remained unanswered among the questions we asked in the introduction. "How does compression affect the alignment accuracy?" In order to answer this, we calculate a correlation between the scores of each possible mapping in compressed domain and the scores that we obtain for these mappings from the original SubMAP method. We consider the scores of each possible subnetwork mapping of compressed nodes found by our framework. Since the mappings found by SubMAP are not of the same form with the mappings in compressed domain, we calculate a score value for each mapping in compressed domain by using the scores of the mappings found by SubMAP. This way, we get two sets of score values one from SubMAP one from our framework for the same set of mappings. We calculate the Pearson's correlation coefficient between these two sets of scores as an indicator of the similarity between the results of the two methods.

Before looking at the correlation values we found, it is important to describe how we calculate a score for a mapping in compressed domain from the mappings of SubMAP. Let P^1 and $\bar{P}^1$ denote the one level compressed forms of two metabolic networks. Let (v_1- $\{\bar{v}_1, \bar{v}_2\}$) denote a mapping in compressed domain where v_1 is a subnetwork of P^1 and $\{\bar{v}_1, \bar{v}_2\}$ is a subnetwork of $\bar{P}^1$. Also, let $v_1 = \{r_1, r_2\}$, $\bar{v}_1 = \{\bar{r}_1, \bar{r}_2\}$ and $\bar{v}_2 = \{\bar{r}_3\}$. We know the edge that maps these two subnetworks has a mapping score in the compressed domain and let us denote it by $|e^1|$ for $c = 1$. We want to compute a mapping score, say $|e|$, for (v_1- $\{\bar{v}_1, \bar{v}_2\}$) from the mappings in original domain that is comparable to $|e^1|$. This subnetwork mapping in compressed domain contains six possible mappings in the original, namely $(r_1, \bar{r}_1)$, $(r_1, \bar{r}_2)$, $(r_1, \bar{r}_3)$, $(r_2, \bar{r}_1)$, $(r_2, \bar{r}_2)$ and $(r_2, \bar{r}_3)$. Let us denote the scores of these mappings in the original domain by $|e_i|$ for $i = 1, 2, ...,6$ respective to their ordering. Then, we compute the mapping score $|e|$ as $\frac{1}{6}\sum_{i=1}^{6} e_i$. It is important to note that, this score is a conservative choice among other possible scoring options. This is because the average can include mapping scores of subnetworks with very low similarities from the original domain of SubMAP. This can underestimate the correct mapping score of $|e|$ and hence degrade the correlation of compressed domain and original domain mapping scores. Overall, for each mapping in compressed domain with a score $|e^c|$ and we calculate the corresponding score $|e|$ in the original domain using this average score.

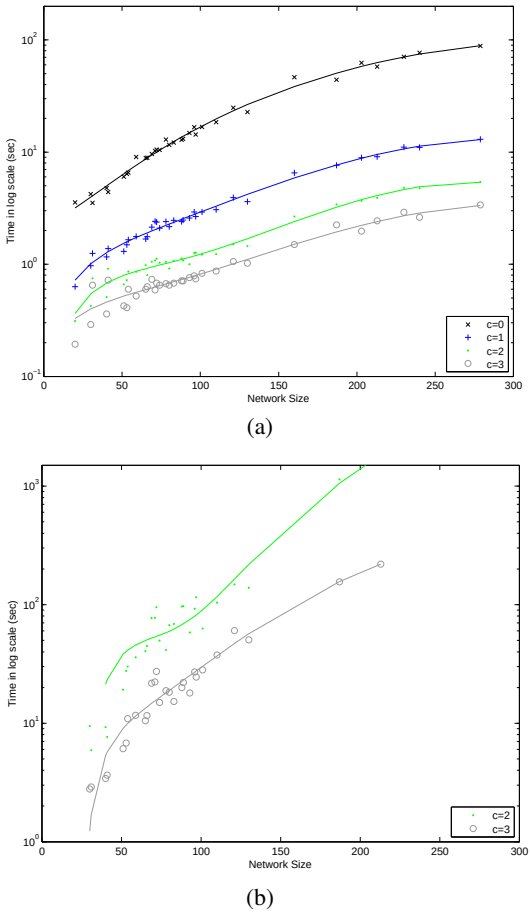

Figure 3: The average running time of our framework when each query network is aligned with all the networks in the selected database set (a) when k = 1 and (b) when k = 2. x-axis is the network size in terms of the number of reactions. $c = 0$ denote the alignments performed with no compression. $c = 1, 2, 3$ denote the results of our framework that compresses both the query and the database networks by c levels before aligning them.

Table 3 summarizes the correlation values found from a total of 3600 alignments. For each of the nine combinations of $k = 1, 2, 3$ and $c = 1, 2, 3$, we ran 400 alignments in the compressed domain and calculated the correlation of each with the alignment that has the same k value but is in the original domain (i.e., $c = 0$). Table 3 shows the average correlation values of these 400 alignments for each k value, c value combination. The first column indicates that the alignment found by using only one compression level is highly similar to the alignment using the base method. Combining this with the running time gain in Figure 3(a) for $c = 1$, we can strongly argue that compression by one level not only provides significant improvement in running time but also accurately captures very high percent of the original alignment results. The accuracy measured in terms of correlation drops to 0.57 on the average when

Table 3: ***Correlation of the mapping scores found by SubMAP and by our framework.***

k/c	1	2	3	Average
1	0.89	0.56	0.53	0.66
2	0.85	0.58	0.50	0.64
3	0.84	0.57	0.49	0.64
Average	0.86	0.57	0.51	0.65

we perform the second level of compression and to 0.51 for the third level. *These results suggest that we can almost always use one level of compression to benefit from a high performance gain without losing much accuracy in terms of the alignment results. For $c = 2$ and $c = 3$, even though the accuracy of their results are significantly better than random, they should be used with caution if the accuracy of the alignment is the main concern.*

5. CONCLUSION

In this paper, we considered the problem of aligning two metabolic networks particularly when both of them are too large to be dealt with using existing methods. To solve this problem, we developed a framework that scales the size of the metabolic networks that existing methods can align significantly. Our framework is generic as it can be used to improve the scalability of any existing network alignment method. It has three major phases, namely the *compression phase*, the *alignment phase* and the *refinement phase*. For the first phase, we developed an algorithm which transforms the given metabolic networks to a compressed domain where they are summarized using much fewer nodes, termed supernodes, and interactions. In the second phase, we carried out the alignment in the compressed domain using an existing method, SubMAP for this paper, as the base alignment algorithm. In the refinement phase, we considered each individual mapping of *supernodes* one by one. Each such mapping corresponds to a smaller instance of network alignment. For each of these mappings, we solved the alignment problem using SubMAP as our base method. Our experiments on the metabolic networks extracted from the KEGG pathway database demonstrate that our compression method reduces the number of reactions by almost half at each level of compression. As a result of this compression, we observe that SubMAP coupled with our framework can align twice or more as large networks as its original version can using the same amount of resources. Our results also suggested that the alignment obtained by only one level of compression benefits from a significant performance gain while capturing the original alignment results with very high accuracy. We believe that this paper takes an important step in scaling the metabolic network alignment problem to real sized networks, and thus, it will have great impact on making the existing computational network alignment methods useful for domain scientists.

6. ACKNOWLEDGMENTS

This work was supported partially by NSF under grants IIS-0845439 and CCF-0829867.

References

[1] F. Ay and T. Kahveci. SubMAP: Aligning metabolic pathways with subnetwork mappings. In *International Conference on Research in Computational Molecular Biology (RECOMB)*, volume LNCS-6044, pages 15–30, 2010.

[2] F. Ay, T. Kahveci, and V. Crecy-Lagard. A fast and accurate algorithm for comparative analysis of metabolic pathways. *Journal of Bioinformatics and Computational Biology (JBCB)*, 7(3):389–428, 2009.

[3] F. Ay, M. Kellis, and T. Kahveci. SubMAP: Aligning metabolic pathways with subnetwork mappings. *Journal of Computational Biology (JCB)*, 18(3):1–17, 2011.

[4] M. Chen and R. Hofestadt. PathAligner: metabolic pathway retrieval and alignment. *Appl Bioinformatics*, 3(4):241–52, 2004.

[5] Q. Cheng, R. Harrison, and A. Zelikovsky. MetNetAligner: a web service tool for metabolic network alignments. *Bioinformatics*, 25(15):1989–90, 2009.

[6] B. Chor and T. Tuller. Biological Networks: Comparison, Conservation, and Evolution via Relative Description Length. *Journal of Computational Biology*, 14(6):817–838, 2007.

[7] T. Dandekar, S. Schuster, B. Snel, M. Huynen, and P. Bork. Pathway alignment: application to the comparative analysis of glycolytic enzymes. *Biochemistry Journal*, 343:115–124, 1999.

[8] B. Dost, T. Shlomi, N. Gupta, E. Ruppin, V. Bafna, and R. Sharan. QNet: A Tool for Querying Protein Interaction Networks. In *International Conference on Research in Computational Molecular Biology (RECOMB)*, pages 1–15, 2007.

[9] C. Francke, R. J. Siezen, and B. Teusink. Reconstructing the metabolic network of a bacterium from its genome. *Trends in Microbiology*, 13(11):550–558, 2005.

[10] M. L. Green and P. D. Karp. A Bayesian method for identifying missing enzymes in predicted metabolic pathway databases. *BMC Bioinformatics*, 5:76, 2004.

[11] H. Jeong, B. Tombor, R. Albert, Z. N. Oltvai, and A. L. Barabasi. The large-scale organization of metabolic networks. *Nature*, 407(6804):651–4, 2000.

[12] M. Kalaev, V. Bafna, and R. Sharan. Fast and accurate alignment of multiple protein networks. *Journal of Computational Biology*, 8:989–99, 2009.

[13] M. Koyuturk, A. Grama, and W. Szpankowski. Pairwise Local Alignment of Protein Interaction Networks Guided by Models of Evolution. In *International Conference on Research in Computational Molecular Biology (RECOMB)*, pages 48–65, 2005.

[14] O. Kuchaiev, T. Milenkovic, V. Memisevic, W. Hayes, and N. Przulj. Topological network alignment uncovers biological function and phylogeny. *Journal of the Royal Society Interface*, 7:1341–1354, 2010.

[15] Y. Li, D. Ridder, M. J. L. de Groot, and M. J. T. Reinders. Metabolic pathway alignment between species using a comprehensive and flexible similarity measure. *BMC Systems Biology*, 2(1):111, 2008.

[16] Z. Li, S. Zhang, Y. Wang, X. S. Zhang, and L. Chen. Alignment of molecular networks by integer quadratic programming. *Bioinformatics*, 23(13):1631–1639, 2007.

[17] C. Liao, K. Lu, M. Baym, R. Singh, and B. Berger. IsoRankN: spectral methods for global alignment of multiple protein networks. *Bioinformatics*, 25(12):253–238, 2009.

[18] H. Ogata, W. Fujibuchi, S. Goto, and M. Kanehisa. A heuristic graph comparison algorithm and its application to detect functionally related enzyme clusters. *Nucleic Acids Research*, 28:4021–8, 2000.

[19] H. Ogata, S. Goto, K. Sato, W. Fujibuchi, H. Bono, and M. Kanehisa. KEGG: Kyoto Encyclopedia of Genes and Genomes. *Nucleic Acids Research*, 27(1):29–34, 1999.

[20] R. Y. Pinter, O. Rokhlenko, E. Yeger-Lotem, and M. Ziv-Ukelson. Alignment of metabolic pathways. *Bioinformatics*, 21(16):3401–8, 2005.

[21] R. Singh, J. Xu, and B. Berger. Global alignment of multiple protein interaction networks with application to functional orthology detection. *Proceedings of the National Academy of Sciences (PNAS)*, 105:12763–8, 2008.

[22] P. Sridhar, T. Kahveci, and S. Ranka. An iterative algorithm for metabolic network-based drug target identification. In *Pacific Symposium on Biocomputing (PSB)*, volume 12, pages 88–99, 2007.