The Eleventh International Symposium on Combinatorial Search
(SoCS 2018)

Focused SANA: Speeding Up Network Alignment

Ilia Leybovich, Rami Puzis, Roni Stern, Maor Reuben
Ben Gurion University of the Negev, Be'er Sheva, Israel
ilialey@post.bgu.ac.il, puzis@bgu.ac.il, sternron@bgumail.bgu.ac.il, maorreu@post.bgu.ac.il

Abstract

Network Alignment (NA) is a generalization of the graph isomorphism problem for non-isomorphic graphs, where the goal is to find a node mapping as close as possible to isomorphism. Recent successful NA algorithms follow a search-based approach, such as simulated annealing. We propose to speed up search-based NA algorithms by pruning the search-space based on heuristic rules derived from the topological features of the aligned nodes. We define several desirable properties of such pruning rules, analyze them theoretically, and propose a pruning rule based on nodes' degrees. Experimental results show that using the proposed rule yields significant speedup and higher alignment quality compared to the state of the art. In addition, we redefine common NA objective functions in terms of established statistical analysis metrics, opening a wide range of possible objective functions.

Introduction

Network Alignment (NA) is a generalization of the graph isomorphism problem for non-isomorphic graphs, where the goal is to find a node mapping, as close to isomorphism as possible. NA is both a problem domain and a component of many algorithms in the Artificial Intelligence (AI) field, see a survey by (Frank, et al., 2016). Finding NA is useful in many AI problem domains including pattern recognition (Conte, et al., 2004), image recognition (Hsieh & Hsu, 2004), NLP (Bayati, et al., 2013), ontology mapping (Li, et al., 2009), and bio-informatics (Elmsallati, et al., 2016). State of the art methods for finding NA employ network embedding (Liu, et al., 2016) heuristic search techniques, e.g., simulated annealing (Mamano & Hayes, 2017), genetic algorithms (Clark & Kalita, 2015; Saraph & Milenković, 2014).

SANA is a state of the art NA method based on simulated annealing that was shown to be superior to many alternative methods in almost all parameters (Mamano & Hayes, 2017; Kanne & Hayes, 2017). Unfortunately, the NA search space grows exponentially with the number of nodes in the aligned networks, which limits the scalability of SANA. Thus, SANA and similar local search algorithms may not be applicable to large networks with hundreds of thousands or millions of nodes.

The primary contribution of this paper is a theory for effective pruning of the NA search space. This pruning focuses the search on alignments that are more likely to be valuable and ignoring alignments where dissimilar nodes are mapped to each other. We present a sufficient condition for such pruning rules that maintains a connected search space. Then, we propose a heuristic pruning rule that satisfies this condition and show empirically that it improves the performance of SANA in both speed and accuracy. For example, on a benchmark of biological networks, the proposed algorithm found a better alignment than SANA in a third of the time.

Beyond improving the search process, we also provide a deeper understanding of the NA objective functions. Finding an appropriate objective function is a hot topic in NA research and a range of objective functions have been proposed in recent years (Kanne & Hayes, 2017). Many of them are ad-hoc and their correctness and effectiveness are only analyzed experimentally. We reframe the NA problem in terms of a prediction problem. This allows importing well-established prediction quality metrics to the field of NA, along with their theoretical guarantees and background.

Problem Definition and Background

There are several definitions of the Network Alignment (NA) problem. We focus on what is known as a *pairwise global alignment*, defined as follows. Let $G_1(V_1, E_1)$ and G_2-(V_2, E_2) be two undirected and unweighted graphs (networks), where V_i and E_i denote the nodes and edges respectively and $|V_1| \leq |V_2|$. A pairwise global alignment between G_1 and G_2 is an injective function $a: V_1 \rightarrow V_2$. Let a^* denote the ideal, unknown, alignment function. The NA problem we address in this paper is to find an alignment that is as close as possible to a^*. A common measure for evaluating an alignment a is *node correctness*, which is the percentage of correctly aligned nodes. That is

$$NC(a) = \left(\sum_{v \in V_1} \delta_{a(v),a^*(v)}\right) / |V_1|$$

where $\delta_{a(v),a^*(v)}$ is the Kronecker delta equal to 1 if $a(v) = a^*(v)$ and zero otherwise. We focus on pairwise global alignments as many prior works do, e.g. (Kuchaiev, et al., 2010; Mamano & Hayes, 2017; Patro & Kingsford, 2012).

A search-based approach for NA was shown to be very effective in practice (Mamano & Hayes, 2017; Clark & Kalita, 2015; Vijayan, et al., 2015). In a search-based NA method, the NA problem is formulated as a search problem where states represent possible alignments, and the NA problem becomes the problem of finding a state that maximizes some objective function. Search-based NA algorithms face two challenges: defining the objective function to evaluate the quality of an alignment, and devising a search strategy efficient enough to handle the huge search space of possible alignments. We elaborate on both topics next.

NA Objective Functions

Search based NA algorithms can optimize various objective functions (Mamano & Hayes, 2017). Ideally, we would use NC as an objective function. However, to compute NC we need the ideal alignment a^*, which is obviously not available in practice. Therefore, search-based algorithms try to optimize other objective functions as a proxy for NC. One of the most successful objective functions is S^3 (Saraph & Milenković, 2014), defined as follows. Let E_a be the set of edges in E_1 that are preserved by a, i.e., $E_a = \{(v, v') \in E_1 | (a(v), a(v')) \in E_2\}$, and let $\hat{E}_a$ be the set of edges in the subgraph of G_2 induced by a, i.e., $\hat{E}_a = \{(u, u') \in E_2 | \exists\ v, v' \in V_1 \wedge a(v) = u \wedge a(v') = u'\}$. The S^3 objective function is defined as $S^3(a) = |E_a| / (|E_1| + |\hat{E}_a| - |E_a|)$. This score punishes not only for edges in E_1 that were left unmatched, but also for edges in E_2 that were supposed to be matched but were not.

Simulated Annealing Network Aligner (SANA)

Given an objective function, one can theoretically use any search algorithm to find an alignment that optimizes this objective function. However, the number of possible alignments is very large (factorial of the number of nodes in the aligned networks), and therefore heuristic search algorithms have been used. Recent work proposed the SANA algorithm (Mamano & Hayes, 2017), which uses the simulated annealing local search algorithm (Kirkpatrick, et al., 1983). SANA was shown to be extremely effective in both runtime and NA quality.

SANA uses SA as follows. The initial state a_0 is a given initial alignment. When such an alignment is not given, SANA chooses a random alignment as a_0. The state transitions are all the alignments that can be created by performing a single alignment manipulation action. SANA allows two types of alignment manipulation actions:

- **Change.** Changing the target of a single node v.
- **Swap**. Swapping the targets of two nodes u, v.

Two alignments are considered *neighbors* if we can obtain one by a single alignment manipulation action on the other. The temperature schedule used by SANA is an exponential decay function (see (Mamano & Hayes, 2017) for details). SANA was shown to be very effective in practice across a range of objective functions. In fact, Mamano and Hayes (2017) consider investigation of effective objective functions for search-based NA as the primary direction for further research.

Limitations of SANA

Although SANA has many advantages, it also has a few shortcomings. First, SANA does not scale well to very large networks. This is because the size of its search space is a factorial of the number of nodes in the aligned networks $(|V_2|!/(|V_2| - |V_1|)!)$, and the branching factor which is $(|V_2| + 0.5|V_1|(|V_1| - 1))$, grows quadratically with the size of G_1. Second, SANA is blind to node properties that are not encompassed by the objective function. Thus, if additional knowledge about the correct alignment is available, SANA cannot consider it without merging it into a single objective function, which is not always possible. In addition, the creators of SANA observed that there are many different alignments with close to perfect S^3 score but far from perfect NC score (Mamano & Hayes, 2017). The NA algorithm we propose addresses these shortcomings.

Focused SANA

Next, we present Focused SANA (F-SANA), an improvement over SANA. The key difference between SANA and F-SANA is two-fold. First, F-SANA creates an intelligent initial alignment. Second, it imposes additional restrictions over the state transitions that can be applied at every iteration of SANA. Informally, these restrictions verify that nodes mapped to each other in any considered alignment are *similar,* where nodes similarity can be computed based on their properties or based on network topology. Using these restrictions drastically reduces the size of the search space, and consequently improves the algorithm's running time. Also, we show that these restrictions result in finding better alignments.

To formally describe the F-SANA algorithm, we introduce the following definitions.

Definition 1 [Ranking Function, Rank]: *For a network* $G(V, E)$, $Rank: V \rightarrow \mathbb{N}$ *is called the Ranking Function, and* $Rank(v)$ *is called the Rank of the node* v.

The ranking induced by a ranking function, may be application dependent (e.g. for social networks it might be address, age etc.), or application independent (e.g. node's degree, betweenness etc.). A *perfect ranking* function for a given NA problem gives the same rank only to nodes that are aligned to each other in a^*, i.e. for all $v \in V_1$ and $u \in V_2$ it holds that $(Rank(v) = Rank(u)) \Leftrightarrow (a^*(v) = u)$. Finding a perfect ranking function is hard. For an imperfect ranking function, the ranks of u and v, such that $a^*(v) = u$, may differ. To this end, we define a *candidate function* (CF) that defines for every rank r a set of ranks ($CF(r) \subset \mathbb{N}$). We expect nodes with rank r to be aligned by a^* to nodes with ranks in $CF(r)$.

Definition 2 [Candidate Function]: *For two networks $G_1(V_1, E_1)$ and $G_2(V_2, E_2)$, a Candidate Function $CF: Rank(V_1) \rightarrow powerset\big(Rank(V_2)\big)$ is a function that maps a rank to a set of ranks.*

For a given node $v \in V_1$, we call the set of nodes in V_2 with a rank in $CF(Rank(v))$ the set of candidates of v. We say that NA is legal if each node in the source network is aligned to one of its candidates in the target network. Formally:

Definition 3 [Legal NA]: *Given NA $a: V_1 \rightarrow V_2$ is Legal if $\forall v \in V_1: Rank(a(v)) \in CF(Rank(v))$.*

The NA algorithm we propose, called F-SANA, uses a given candidate function to limit SANA to consider only alignment manipulation operations that result in a legal alignment. This is done by modifying the random state transition choice as follows.

Let a be the current alignment. For a change operation, we choose a random node v from V_1. Then, we choose a random node from the "unoccupied" candidates of v, and the new alignment maps v to this candidate. For a swap operation, after choosing a random node v from V_1, we choose a random node u from the candidates of v that has an origin in V_1 in the current alignment, i.e., $\exists v' \in V_1$ such that $a(v') = u$. The swap operation is only allowed if $a(v)$ is a candidate of v'. This can be implemented efficiently with appropriate data structures. Note that, for the trivial ranking function that maps all nodes in both networks to 1, F-SANA is exactly SANA. Thus, F-SANA is, in fact, a generalization of SANA. As we discuss below and show experimentally, it is possible to define a non-trivial candidate function that causes F-SANA to perform significantly better than SANA.

Effective candidate functions

The effectiveness of F-SANA depends on the chosen candidate function. We suggest three desirable properties for a candidate function:

(1) **Completeness preserving (CP)**. A candidate function is completeness preserving if every legal alignment can be reached from every other legal alignment by using only the state transition functions allowed by F-SANA. That is, the search space of F-SANA is connected when using this candidate function.

(2) **Efficient.** A candidate function is efficient if its computation time is lower than the time saved during the search process due to the search space pruning.

(3) **Optimality preserving (OP)**. A candidate function is optimality preserving if a^* is legal w.r.t. its ranking function.

In addition, the running time of F-SANA depends on the number of candidates for every node. Thus, it is desirable for an effective candidate function to map as few candidates as possible for each node. However, there is a tradeoff, since we need to be careful to stay as close as possible to being OP.

Conditions for a CP candidate function

Some candidate functions are not CP (an example was omitted due to space limitations).

Next, we propose a sufficient condition for identifying that a given candidate function is CP. Checking whether it is also efficient requires a runtime complexity analysis. We leave providing sufficient conditions for OP to future research.

Theorem 1: *Let $G_1(V_1, E_1)$, $G_2(V_2, E_2)$, be networks such that $|V_1| \leq |V_2|$, with a ranking function $Rank$ and a candidate function CF. If (*) $\forall X \in Img(CF)$: X is continuous (i.e. $\forall i, j \in X: i \leq h \leq j \rightarrow h \in X$), then every 2 legal alignments are connected in the space of legal NAs.*

Proof is omitted due to space constraints.

For example, consider two social networks where each node represents a person, and is associated with that person's age and gender. Now assume we want to align these networks so that nodes that represent the same person are aligned to each other. Assuming that the edge and gender are correctly reported in both social networks, we can build a perfect ranking function by marking nodes with the same age and gender by the same number with respect to lexicographical order of age and gender. Obviously this candidate function is CP by Theorem 1. It is efficient since $O(|V|)$ is much smaller than the complexity of the search part of the algorithm. It is also OP (assuming that the data is correct and the networks consist of the same people).

NA Objective Functions

F-SANA improves on SANA by focusing its search using the candidate functions. A key factor in the success of search algorithms is the objective functions they aim to optimize. In fact, (Mamano & Hayes, 2017) claimed that the research for better objective functions should be the main focus of NA research. Indeed, many objective functions have been proposed for NA. To choose in a principle manner the appropriate objective function, we map in this section common NA objective functions to classical measures from the Machine Learning (ML) literature for evaluating classifiers.

A binary ML classifier assigns either Positive (P) or Negative (N) label to any given instance (i.e. an object being classified). When the classifier is evaluated the predicted values are compared to the ground truth. Instances that are labelled by the classifier as Positive and are indeed positive according to the ground truth are called True Positive (TP) instances. Instances that are incorrectly classified as Positive by the classifier are called False Positive (FP) instances. True Negative (TN) and False Negative (FN) instances are defined symmetrically.

For the sake of the following discussion consider a classifier $cl: V_1 \times V_1 \rightarrow \{P, N\}$ which assigns a Positive (P) label to a pair of nodes if they are adjacent in G_1 and a Negative (N) label otherwise:

$$cl((v_1, v_2)) = \begin{cases} P, & (v_1, v_2) \in E_1 \\ N, & Otherwise \end{cases}$$

We say that an alignment a maps an edge $(v_1, v_2) \in E_1$ to an edge in G_2 iff $\left(a(v_1), a(v_2)\right) \in E_2$. We expect all edges in G_1 to be mapped to edges in G_2 and all non-adjacent pairs of nodes in G_1 to be mapped to non-adjacent pairs of nodes in G_2. Thus the classifier cl assigns the correct label for (v_1, v_2) iff $(v_1, v_2) \in E_1 \Leftrightarrow \left(a(v_1), a(v_2)\right) \in E_2$.

Next we define the confusion matrix (TP, FP, TN, FN) in terms of NA. The TP is the set of edges in $E_1 \subseteq V_1 \times V_1$ (hence positive) that were mapped by a to edges in E_2 (hence true): $TP = \left\{(v, v') \in E_1 : \left(a(v), a(v')\right) \in E_2\right\}$.

FP is the set of edges in E_1 that were mapped to non-edges in G_2: $FP = \left\{(v, v') \in E_1 : \left(a(v), a(v')\right) \notin E_2\right\}$.

FN are the pairs of nodes which are not edges in G_1 (hence negative) that were mapped to edges of G_2 (hence false):

$FN = \{(v, v') \notin E_1 : \left(a(v), a(v')\right) \in E_2\}$

Finally, TN is the set of non-edges that were mapped to non-edges.

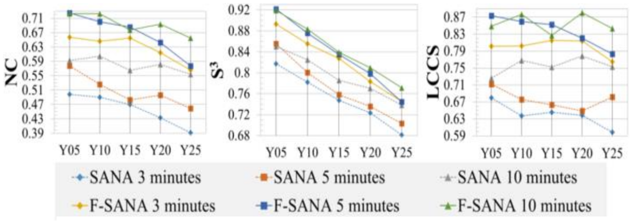

Fig. 1. *Results for the second dataset.*

These observations allow representing many NA objective functions in the same terms in which classification performance measures are defined (e.g. precision, accuracy, recall). Moreover, we show (omitted due to space restrictions) that some objective functions suggested in literature are "re-invented" measures, e.g. $EC(a)$ by (Kuchaiev, et al., 2010) & $ICS(a)$ by (Patro & Kingsford, 2012) are actually precision & recall respectively. Also the score $S^3(a) = |TP|/(|TP| + |FP| + |FN|)$ is very similar to the well known F_1-score $= 2|TP|/(2|TP| + |FP| + |FN|)$.

Beyond the elegance of mapping NA specific measures to common measures from the ML literature, this mapping opens the door to importing a wide range of more sophisticated objective functions from the ML literature back to NA, leveraging the years of ML research behind them.

Experiments

To show the advantages of F-SANA in practice, we compared it experimentally with SANA. In this paper we present only one set of experiments due to space constraints.

As a ranking function, we used the nodes' degrees. We experimented with several candidate functions, and ended up using the following. First, we sorted the node of each network by their degrees. Let $v_1, \ldots, v_n$ and $u_1, \ldots, u_m$ be the nodes, where $\deg(v_i) \leq \deg(v_{i+1})$ and $\deg(u_i) \leq \deg(u_{i+1})$ for every i. Then, we create an initial alignment, denoted a_0, by mapping nodes according to their order, i.e., $a_0(v_i) = u_i$. For every rank r of nodes in V_1 we denote by $R(r)$ the set of nodes in V_2 that were mapped to nodes in V_1 with rank r:

$$R(r) = \{u \in V_2 | \, \exists v \in V_1 : a_0(v) = u \wedge Rank(v) = r\}$$

Let $Ranks(V_2)$ be the set of all ranks in V_2. We defined the candidate function CF as follows

$$CF(r) = \{r' | r' \in Ranks(V_2) \ \wedge \ r' \in \left[m - \lceil\sqrt{m}\rceil, M + \lceil\sqrt{M}\rceil\right]\}$$

Where $m = \min\{r, min\left(R(r)\right)\}$ & $M = \max\{r, \max\left(R(r)\right)\}$.

Intuitively, CF extends the initial alignment so that nodes of the same rank will have the same set of candidates and this set is continuous, hence CF is CP (Theorem 1). Also, we extend the rank bounds by $\lceil\sqrt{m}\rceil$ and $\lceil\sqrt{M}\rceil$.

The dataset we experimented on is a standard NA benchmark from (Collins, et al., 2007), that was used to evaluate SANA and many other algorithms. It consists of 6 networks denoted Y0, Y5, Y10, Y15, Y20, and Y25. Each network represents a protein-protein interaction (PPI) network of yeast. All the networks have 1004 nodes. The difference between the six networks is the number of edges where YX has X% more edges than Y0, which has 8323 edges.

To assess the quality of the alignments generated by SANA and F-SANA, we used the NC (since we know a^*). In addition, we report the S^3 score and the Largest Common Connected Subgraph (LCCS), which indicates whether the alignment found a large similar subgraph (Kuchaiev, et al., 2010; Saraph & Milenković, 2014).

We ran SANA and F-SANA to align the base network Y0 to each of its noisy variants Y05,…,Y25. Figure 1 shows the NC, S^3, and LCCS results as a function of the number of edges added to the base network (x-axis). Different data lines correspond to different algorithms and time budgets (3, 5, and 10 minutes). The same general trend can be observed in these results also: F-SANA is able to find higher quality alignments, according to all evaluated measures (NC, S^3, and LCCS) for every given time budget and network pair, regardless of how many edges were added. Even F-SANA with 3 minutes time budget (yellow solid line with diamond markers) achieves better results than SANA with 10 minutes, in every measure and every network pair. These results show the robustness of F-SANA's advantage over SANA. In conclusion, F-SANA exhibits higher quality network alignments in shorter times compared to SANA over a range of datasets, time budgets, and network variants.

Conclusions

We proposed two contributions to the NA search problem. Primarily, we propose an effective way to perform search space pruning for NA. We provide some theoretical foundation for this pruning and implement this approach into an improved algorithm based on SANA. The proposed pruning approach is applicable to other search based NA methods as well. Additionally, we reframe the NA in terms of classification/prediction in order to significantly expand the set of objective functions (an important research focus in NA) with well-established performance metrics used in statistical analysis. We suggest that research in those two directions will provide great improvement in the ability to align networks, and make NA even more widely applicable in even more domains than it already is.

Acknowledgement

This research was funded by the Israeli Ministry of Defense as a part of the Social Network Digestion project and by the Israeli Science Foundation (ISF) grant no. 210/17 to Roni Stern.

References

Aladaug, A. E. & Erten, C., 2013. SPINAL: scalable protein interaction network alignment. *Bioinformatics.*

Bayati, M., Gleich, D. F., Saberi, A. & Wang, Y., 2013. Message-Passing Algorithms for Sparse Network Alignment. *ACM Transactions on Knowledge Discovery from Data (TKDD).*

Clark, C. & Kalita, J., 2014. A comparison of algorithms for the pairwise alignment of biological networks. *Bioinformatics.*

Clark, C. & Kalita, J., 2015. A multiobjective memetic algorithm for ppi network alignment. *Bioinformatics.*

Collins, S. R. et al., 2007. Toward a comprehensive atlas of the physical interactome of Saccharomyces cerevisiae. *Mol. Cell. Proteomics,* Volume 6, pp. 439-450.

Conte, D., Foggia, P., Sansone, C. & Vento, M., 2004. Thirty years of graph matching in pattern recognition. *International Journal of Pattern Recognition and Artficial Intelligence.*

El-Kebir, M., Heringa, J. & Klau, G. W., 2011. *Lagrangian relaxation applied to sparse global network alignment.* s.l., s.n.

Elmsallati, A., Clark, C. & Kalita, J., 2016. Global alignment of protein-protein inter-action networks: A survey. *IEEE/ACM Transactions on Computational Biology and Bioinformatics.*

Faisal, F. E., Meng, L., Crawford, J. & Milenkovic, T., 2015. The post-genomic era of biological network alignment. *EURASIP Journal on Bioinformatics and Systems Biology.*

Frank, E.-S., Matthias, a. D. & Yongtang, S., 2016. Fifty years of graph matching, network alignment and network comparison. *Information Sciences.*

Guzzi, P. H. & Milenkovic, T., 2017. Survey of local and global biological network alignment: the need to reconcile the two sides of the same coin. *Briefings in bioinformatics.*

Hashemifar, S. & Xu, J., 2014. Hubalign: an accurate and efficient method for global alignment of protein--protein interaction networks. *Bioinformatics.*

Hsieh, S.-M. & Hsu, C.-C., 2004. Graph-based representation for similarity retrieval of symbolic images. *Data & Knowledge Engineering.*

Ibragimov, R., Malek, M., Guo, J. & Baumbach, J., 2013. *GEDEVO: An Evolutionary Graph Edit Distance Algorithm for Biological Network Alignment.* s.l., German Conference on Bioinformatics.

Kanne, D. P. & Hayes, W. B., 2017. *SANA: separating the search algorithm from the objective function in biological network alignment, Part 1: Search,* s.l.: arXiv.org.

Kirkpatrick, S., Gelatt, C. D. & Vecchi, M. P., 1983. Optimization by simulated annealing. *Science.*

Kirkpatrick, S., Gelatt, C. D. & Vecchi, P. M., 1983. Optimization by simulated annealing. *Science,* 220(4598), pp. 671-680.

Kuchaiev, O. et al., 2010. Topological network alignment uncovers biological function and phylogeny. *Journal of the Royal Society Interface.*

Li, J., Tang, J., Li, Y. & Luo, Q., 2009. RiMOM: A dynamic multistrategy ontology alignment framework.. *IEEE Transactions on Knowledge and Data Engineering.*

Liu, L., Cheung, W. K., Li, X. & Liao, L., 2016. *Aligning Users across Social Networks Using Network Embedding.* s.l., IJCAI.

Malek, M., Ibragimov, R., Albrecht, M. & Baumbach, J., 2015. CytoGEDEVO—global alignment of biological networks with Cytoscape. *Bioinformatics.*

Malod-Dognin, N. & Prvzulj, N., 2015. L-GRAAL: Lagrangian graphlet-based network aligner. *Bioinformatics.*

Mamano, N. & Hayes, W. B., 2017. SANA: simulated annealing far outperforms many other search algorithms for biological network alignment. *Bioinformatics.*

Neyshabur, B., Khadem, A., Hashemifar, S. & Arab, S. S., 2013. NETAL: a new graph-based method for global alignment of protein--protein interaction networks. *Bioinformatics.*

Patro, R. & Kingsford, C., 2012. Global network alignment using multiscale spectral signatures.. *Bioinformatics.*

Saraph, V. & Milenković, T., 2014. MAGNA: maximizing accuracy in global network alignment.. *Bioinformatics.*

Sun, Y., Crawford, J., Tang, J. & Milenkovic, T., 2015. *Simultaneous optimization of both node and edge conservation in network alignment via WAVE.* s.l., International Workshop on Algorithms in Bioinformatics.

Vijayan, V., Saraph, V. & Milenkovi'c, T., 2015. MAGNA++: Maximizing Accuracy in Global Network Alignment via both node and edge conservation. *Bioinformatics.*