LOCANDA: exploiting causality in the reconstruction of gene regulatory networks

Gianvito Pio, Michelangelo Ceci, Francesca Prisciandaro and Donato Malerba

Department of Computer Science, University of Bari Aldo Moro
Via Orabona, 4 - 70125 Bari - Italy

gianvito.pio@uniba.it, michelangelo.ceci@uniba.it,
f.prisciandaro1@studenti.uniba.it, donato.malerba@uniba.it

Abstract. The reconstruction of gene regulatory networks via link prediction methods is receiving increasing attention due to the large availability of data, mainly produced by high throughput technologies. However, the reconstructed networks often suffer from a high amount of false positive links, which are actually the result of indirect regulation activities. Such false links are mainly due to the presence of common cause and common effect phenomena, which are typically present in gene regulatory networks. Existing methods for the identification of a transitive reduction of a network or for the removal of (possibly) redundant links suffer from limitations about the structure of the network or the nature/length of the indirect regulation, and often require additional pre-processing steps to handle specific peculiarities of the networks at hand (e.g., cycles).
In this paper, we propose the method LOCANDA, which overcomes these limitations and is able to identify and exploit indirect relationships of arbitrary length to remove links considered as false positives. This is performed by identifying indirect paths in the network and by comparing their reliability with that of direct links. Experiments performed on networks of two organisms (*E. coli* and *S. cerevisiae*) show a higher accuracy in the reconstruction with respect to the considered competitors, as well as a higher robustness to the presence of noise in the data.

Keywords: Causality, Bionformatics, Gene Network Reconstruction.

1 Introduction

Recent studies in biology have been significantly supported by high throughput technologies and by computational methods, which led to an improved understanding of the working mechanisms in several organisms. Such mechanisms can be usually modeled through biological networks, which are able to easily describe the considered biological entities as well as their relationships and interactions. On the basis of the phenomenon under study, different types of biological networks can be considered. The most prominent example is that of networks modeling the control of transcription into messenger RNAs or proteins [2, 13]. In these

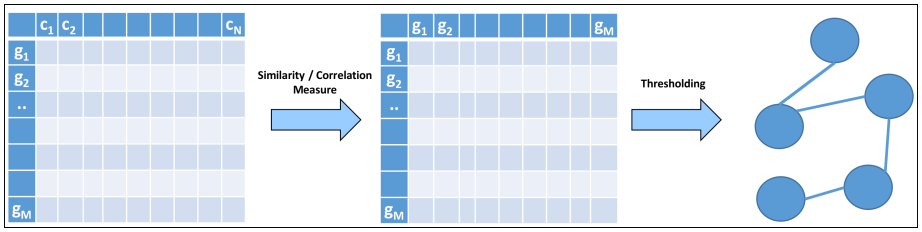


Fig. 1. Network reconstruction from expression data. On the left, a matrix of M genes, each associated to a vector containing the expression level measured under N different conditions. In the middle, the gene-gene matrix obtained by pair-wisely computing a similarity/correlation measure between the vectors. On the right, the reconstructed network obtained by imposing a threshold on the values of the gene-gene matrix.

networks, called Gene-Regulatory Networks (GRNs), nodes represent molecular entities, such as transcription factors, proteins and metabolites, whereas edges represent interactions, such as protein-protein and protein-DNA interactions.

The direct observation of the real structure of these interaction networks would require expensive in-lab experiments, usually performed through the so-called epistasis analysis. Although in the literature we can find some computational approaches which support such an analysis [17], gene expression data are much easier to obtain, therefore most of computational approaches proposed in the literature focused on predicting the existence of interactions from gene expression data, mainly on the basis of link prediction methods. These approaches analyze the expression level of the genes under different conditions (e.g., with a specific disease or after a treatment with a specific drug) or, alternatively, under a single condition in different time instants. The expression levels observed for each gene are represented as a feature vector and a gene-gene matrix is built by pair-wisely computing a similarity, correlation or information-theory-based measure between the vectors associated to genes [6]. Finally, the existence of links is inferred by imposing a threshold on the obtained score (see Figure 1), where the direction is inferred only if the considered measure is asymmetric.

However, except for those based on clustering [15], these methods generally assume the independence among the interactions, i.e., they focus on each pair of genes separately, disregarding possible dependencies or indirect influences among them. This assumption leads to predict false positive interactions, which are usually due to causality phenomena: *i)* common regulator genes (also referred to as *common cause* in the literature [9]) or *ii)* commonly regulated genes (also referred to as *common effect* in the literature [9]). In the first case (see Figure 2(a)), the feature vector associated to a gene C which exhibits a regulatory activity on two genes A and B will presumably be similar to the feature vectors associated to A and B. However, even if there is no interaction between the genes A and B, their feature vectors will appear similar, therefore a link between them could possibly be detected. Analogously, in the second case (see Figure 2(b)), a gene C which is regulated by two genes A and B will presumably have a feature

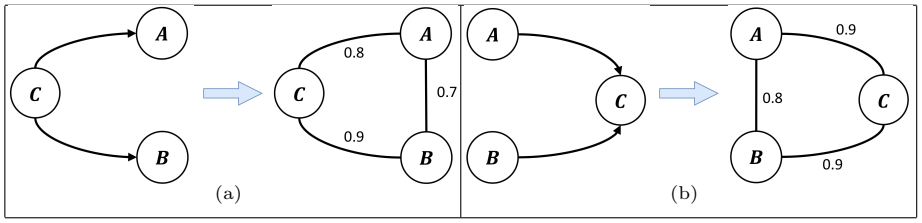


Fig. 2. Issues in the network reconstruction due to common cause (a) or common effect (b) phenomena. The direction of the interactions does not appear in the reconstructed networks if we consider the case of a symmetric similarity/correlation measure.

vector which is similar to the feature vectors associated to A and B. Therefore, even if A and B do not interact, their feature vectors will be similar and a link between them will possibly appear in the reconstructed network. Such issues are even more evident when data are affected by noise. Indeed, possible measurement errors can lead to a significant increment of false positives due to common cause and common effect phenomena, compromising the quality of the reconstruction.

The presence of these phenomena in the reconstruction of gene regulatory networks has been largely recognized in the literature, also considering possible hidden common causes and hidden common effects [10], and several approaches for post-processing the gene-gene matrix have been proposed. These methods, usually called *scoring schemes* [6], analyze large sets of genes simultaneously, in order to catch more global interaction activities and possibly reduce false positives due to the presence of common cause and common effect phenomena. One of the most popular scoring scheme is ARACNE [11], which evaluates all the possible connected gene triplets and removes the edge with the lowest score. ARACNE is limited to undirected networks and is not able to analyze more global indirect interactions (i.e., involving more than three genes). However, although the idea of removing the weaker edge is very simple, the intuition of considering the score as an indication of the reliability of the interaction is reasonable, and has been exploited by other works in the literature (e.g., [3]).

In this paper, inspired by the same idea, we introduce a new method, called LOCANDA, which is able to identify interaction chains of arbitrary length and is able to remove false positive interactions working on the identified chains. It is noteworthy that the approach we propose in this paper has its roots in methods for the analysis of graphs and, in particular, in works for the transitive reduction [1, 7]. However, differently from existing methods, LOCANDA is able to handle weighted, directed and possibly cyclic networks without any pre-processing step.

In the Section 2, we briefly describe existing methods which exploit causality in the analysis or in the reconstruction of networks, giving emphasis to those tailored for the identification and removal of indirect interactions in (biological) networks. In Section 3, we describe our method LOCANDA, while in Section 4 we describe the experiments we performed and comment the obtained results. Finally, in Section 5, we draw some conclusions and outline possible future works.

2 Related Work

In the literature, we can find several approaches which catch and exploit causality phenomena for different goals. A general framework for the identification of causal links between variables [12] consists in *i)* the analysis of correlations, which suggest possible (undirected) links, *ii)* the analysis of partial correlations, which can be exploited to remove possibly indirect relationships and *iii)* some assumptions on the structure of the network, such as acyclicity, which can suggest the possible direction of links. It is noteworthy that such assumptions can be easily violated in specific domains, such as biology, leading to an inaccurate reconstruction. An example of application of such a framework can be found in algorithms for learning the structure of Bayesian networks [9], which identify causalities between variables by analyzing the *d-separation* among them, which is based on the common cause and effect phenomena described in Section 1.

Other approaches exploit the concept of causality to identify a transitive reduction of a graph [1, 7]. These methods analyze a graph and produce a new graph containing a subset of links, which guarantees to convey the same information of the original graph. This means that, analogously to the method proposed in this paper, these approaches aim at removing edges that can be considered the result of an indirect relationship. Specifically, the method proposed in [1] finds a transitive reduction G' of the initial graph G, where G' has a directed path from vertex u to vertex v if and only if G has a directed path from vertex u to vertex v and there is no graph with such a property having fewer edges than G'. In other words, the obtained graph G' is the smallest graph (in terms of edges) such that given any pair of nodes $\langle u, v \rangle$, if v is (respectively, is not) reachable from u in the initial graph G, then v is (respectively, is not) reachable in the reduced graph G'. This means that the information conveyed by the graph, in this work, is associated to the reachability of nodes. Although based on the same principles of LOCANDA, this approach requires the identification of an equivalent acyclic graph before performing the analysis and is limited to unweighted graphs. Therefore, it can not exploit information about the reliability commonly associated to each edge in biological networks.

Analogously, in [7] the authors propose the identification of a Minimal Equivalent Graph (MEG), whose definition is the same as the transitive reduction proposed in [1]. The method consists of several steps, that are: *i)* the identification of strongly connected components, *ii)* the removal of cycles from each component, *iii)* the identification of the minimal equivalent graph for each component and *iv)* the reintroduction of the edges removed in the step *i)*. Even if more sophisticated, this approach suffers from the same limitations described for [1].

Focusing on biological networks, in the literature, several approaches have been proposed to consider specific issues as well to exploit specific characteristics of such an application domain. In particular, it is possible to exploit the causality to infer the directionality of the interactions by exploiting time-series gene expression data [6]. In this case, the regulator gene (the cause), by definition, should act before the regulated gene (the effect). Therefore, a common strategy consists in computing the similarity between two genes u and v, by performing a

progressive shifting forward in time of the time-series associated to the first gene u. If the similarity increases, then it is possible to conclude that u acts before v, therefore u regulates v. More sophisticated approaches exploit Granger causality [8] or hidden (i.e., unobserved, latent) common causes and common effects [10]. All these methods, however, are applicable only when analyzing time-series data, while they cannot be applied when each gene is associated with a vector representing its expression values in different (steady) conditions. In [11], the authors propose the (already mentioned) method ARACNE which exploits causality phenomena to identify and remove indirect relationships. Analogously to the approach presented in this paper, the method acts as a post-processing phase of the network reconstruction, aiming at removing interactions considered as the indirect effect of other interactions. This is performed by analyzing all the triplets of connected genes and by removing the weakest interaction, i.e. the edge with the lowest score. As already clarified, although based on the same principle of LOCANDA, this approach is limited to indirect interactions involving only three genes, thus it cannot identify interaction chains of arbitrary length.

In a more recent work [3], the authors propose a method for the identification of the transitive reduction of biological networks. This method is able to analyze both unweighted networks and possibly cyclic weighted networks. In this last case, however, following the approach adopted in [14], it requires to pre-process the network in order to make it acyclic. In detail, the method *i)* identifies and shrinks the strongly connected components into single nodes, *ii)* applies the reduction on the resulting acyclic graph, and *iii)* re-expands the components. It is noteworthy that this procedure assumes that genes within each component are fully connected and do not perform any reduction within each component, since the results would strongly depend on the order of the analysis. Moreover, it assumes that the graph resulting from the step *i)* is acyclic, i.e., there is no cycle among the components. However, the reduction phase is based on an idea which is similar to that adopted in LOCANDA, i.e. on the computation of an uncertainty score for paths connecting nodes, and on the removal of direct links having a higher uncertainty with respect to the identified indirect paths.

In summary, with respect to existing works in the literature, the method LOCANDA proposed in this paper identifies and removes links which are considered as the result of indirect regulation activities, exploiting common cause and common effect phenomena. LOCANDA has the following distinguishing characteristics: *i)* unlike classical methods for the identification of a transitive reduction of networks [1, 7], it is able to work on weighted networks, which is relevant when dealing with reconstructed biological networks where edges are associated to a score/reliability; *ii)* unlike [11], it is able to work on directed networks, which (if available) becomes important to correctly consider causality phenomena; *iii)* similar to [3] and unlike [11], it is able to catch indirect relationships of arbitrary length by comparing the reliability of direct links to that of identified indirect relationships; *iv)* contrary to [1], [3] and [7] it is able to directly work on cyclic networks, without any pre-processing steps and by guaranteeing the same result independently on the order of analysis.

3 The method LOCANDA

In this section, we describe the method LOCANDA for the identification and removal of false positive links in a reconstructed gene network. The method is based on the concepts of common cause and common effect already introduced in Section 1. We remind that LOCANDA is not limited to the simple cases depicted in Figure 2, but is able to detect and exploit indirect relationships of arbitrary length. In the following, before describing our approach, we introduce some useful notions and formally define the task we solve. Let:

- V be the set of genes, i.e., nodes in the reconstructed network.
- $E \subseteq (V \times V \times \mathbb{R})$ be the set of interactions in the reconstructed network, i.e., weighted edges in the form $\langle source\_node, destination\_node, edge\_weight \rangle$.
- P be a generic path between two nodes v_1 (*source node*) and v_k (*destination node*) in the network, defined as a sequence of nodes $[v_1, v_2, \ldots, v_k]$, such that $\forall_{i=1,2,\ldots,k-1}, \exists w_i : \langle v_i, v_{i+1}, w_i \rangle \in E$.
- $f(P)$ be a function that measures the reliability of the path P according to the edges involved in its sequence of nodes.

A path P between u and v is considered more reliable than the edge $\langle u, v, w \rangle$ if $f(P) > w$. According to such an assumption, the task we solve consists in the identification of a reduced set of edges $\widetilde{E} \subseteq E$, satisfying the following properties:

- the *reachability* of nodes is preserved. Formally, given two nodes u, v, there exists at least a path P connecting them through the edges in $\widetilde{E}$ if and only if there exists at least a path P connecting them through the edges in E.
- an edge $\langle u, v, w \rangle$ is removed, i.e., it does not belong to the reduced set $\widetilde{E}$, if there exists a path P from u to v which is more reliable than $\langle u, v, w \rangle$.

Note that, contrary to [1] and [7], we do not require the minimality of the number of edges in the reduced network, since we are not interested in *pure* transitive reduction, but in removing possible false positive edges identified during the reconstruction of the network. Indeed, in the case of reconstructed gene networks, the fact that the information conveyed by a link can be represented by a sequence of nodes (a path) is not a sufficient condition to consider the link as a false positive due to the presence of common cause or common effect phenomena. For this reason, we remove a link only if its reliability appears lower than the reliability of the identified path, measured by $f(\cdot)$. In this work, we take into account different possible measures to estimate the reliability of the path. In particular, being $w(v_i, v_j)$ the weight associated to the edge between v_i and v_j, we consider the following measures:

- *Minimum (Min)*, which corresponds to the lowest edge weight in the path, following the principle of the "weakest link in the chain".
  Formally, $f([v_1, v_2, \ldots, v_k]) = \min_{i=1,2,\ldots,k-1} w(v_i, v_{i+1})$.

- *Product (Prod)*, i.e., the product of the edge weights involved in the path. This approach is motivated by the common strategy adopted for the combination of probabilities of (naively independent) events.
  Formally, $f([v_1, v_2, \ldots, v_k]) = \prod_{i=1}^{k-1} w(v_i, v_{i+1})$.
- *Average (Avg)*, i.e., the average of the edge weights involved in the path. Formally, $f([v_1, v_2, \ldots, v_k]) = \frac{1}{k} \cdot \sum_{i=1}^{k-1} w(v_i, v_{i+1})$.
- *Weighted Average (WAvg)*, i.e., the average of the edge weights involved in the path, linearly weighted on the basis of their closeness to the source node. This approach can be motivated by the assumption that the influence of the source node on the other nodes in the path fades linearly on the basis of their distance. Formally, $f([v_1, v_2, \ldots, v_k]) = \frac{1}{\sum_{i=1}^{k-1} \frac{1}{i}} \cdot \sum_{i=1}^{k-1} \left[\frac{1}{i} \cdot w(v_i, v_{i+1})\right]$.

The pseudo-code of the algorithm LOCANDA is reported in Algorithm 1. We also report a running example in Figure 3. Before describing LOCANDA, we remind that the method is able to analyze both undirected and directed networks, weighted according to a score representing the reliability about the existence of the interaction (computed by any method for network reconstruction). Here we assume to work with a weighted directed network (the most general case), since an unweighted network can be always mapped into a directed network by introducing an edge for each direction, with the same reliability score. The first step of LOCANDA consists in the removal of self-edges (line 2), since some methods for network reconstruction identify them erroneously. Although self-regulation activities are possible in biology, in reconstructed networks such links are due to errors in the computation of similarity/correlation measures on the vector associated to a single gene. In our example, the self-edge on the node E (Figure 3(b)) is removed, leading to the network in Figure 3(c). Then the algorithm analyzes each node (that we call *source node*) aiming at identifying all the reachable nodes and a path to reach them. Note that the visit of the network is performed according to a depth-first and best-first strategy, based on the reliability of the edges. The algorithm works in a greedy fashion, since an exhaustive exploration of all the possible paths would lead to an exponential time complexity. When there are several edges to follow, LOCANDA considers the path that locally (i.e., by observing only the neighborhood) appears the most reliable.

LOCANDA exploits three data structures: the set of visited nodes (*visited*), the current sequence of nodes (*path*) and a *stack*, according to which nodes are explored. Moreover, it exploits a structure (RT) similar to the routing table used by routing algorithms, which keeps information about the nodes reachable from the source node. In particular, for each reachable node (*destination*), it stores:

- the **next-hop**, i.e., the node adjacent to the source node that we need to follow to reach it, according to the current path.
- the **path score** associated to the current path, on which is based the choice of the optimal path to keep. LOCANDA will prefer a new path with respect to a previously identified path if this value is higher.
- the **path weight**, which represents the reliability associated to the current path according to $f(\cdot)$, that will be exploited to remove links.

Algorithm 1: Pseudo-code of the method LOCANDA.

Data:
- $\cdot V$: the set of genes (nodes in the network)
- $\cdot E \in (V \times V \times \mathbb{R})$: the set of interactions (edges in the network), represented as $\langle source\_node, destination\_node, edge\_weight \rangle$
- $\cdot f(\cdot)$: the measure for the reliability of a path

Result:
- $\cdot \widetilde{E}$: the updated (reduced) set of interactions

```
1  begin
2      Ẽ ← E \ E.getSelfEdges();
3      foreach src ∈ V do
           /* Structures initialization. Records in the routing table RT are in the form
              ⟨dest_node, next_hop, path_score, path_weight⟩. Operations on RT are based on
              dest_node. Updates are considered as a new record if it does not exist.   */
4          visited ← {src}; path ← [src]; path_score ← 0; stack ← [ ]; RT ← [ ];
           /* Initialize the routing table for adjacents of src                          */
5          foreach ⟨src, adj, w⟩ ∈ Ẽ in ascending order w.r.t. w do
6              RT.update(adj, adj, w, f([adj]));
7              stack.push(adj);
8          while stack is not empty do
9              current_node ← stack.pop();
10             visited ← visited ∪ {current_node};
11             edge_weight ← Ẽ.getEdgeWeight(path.getLast(), current_node);
12             old_path_score ← RT.getPathScore(current_node);
13             new_path_score ← path_score + edge_weight;
               /* Update the RT if the route does not exist or if the new path has a
                  higher score than the previous path                                   */
14             if old_path_score = null or old_path_score < new_path_score then
15                 next_hop ← path.getFirst();
16                 RT.update(current_node, next_hop, new_path_score, f(path));
               /* Push non-visited adjacent nodes of the current node into the stack,
                  ordered by weight                                                     */
17             foreach ⟨current_node, adj, w⟩ ∈ Ẽ in ascending order w.r.t. w do
18                 if adj ∉ visited then
19                     stack.push(adj);
               /* Update the current path                                               */
20             if some nodes were added to stack then
21                 path.add(current_node);
22                 path_score ← new_path_score;
23             else if stack is not empty then
24                 next ← stack.top();
25                 while ⟨path.getLast(), next⟩ ∉ Ẽ do
26                     last ← path.getLast();
27                     path.removeLast();
28                     path_score ← path_score − Ẽ.getEdgeWeight(path.getLast(), last);
           /* Remove a direct link if it is not used to reach other nodes and its less
              reliable than the indirect link (path)                                    */
29         all_next_hops ← RT.getAllNextHops();
30         foreach ⟨src, adj, w⟩ ∈ Ẽ do
31             if adj ∉ all_next_hops and w < RT.getPathWeight(adj) then
32                 Ẽ ← Ẽ \ {⟨src, adj, w⟩};
33     return Ẽ;
```

Note that we prefer to consider two different criteria for the choice of the optimal path to consider (path score) and for the estimation of the reliability of the path (path weight), since they could not be generally based on the same assumptions. In particular, the path score will correspond to the sum of edges in the path, since, combined with the adopted strategy for the choice of the edge to follow (i.e., the highest), leads to the identification of long and reliable paths. On the contrary, the estimation of the path weight will be based on several different measures, that we will describe later.

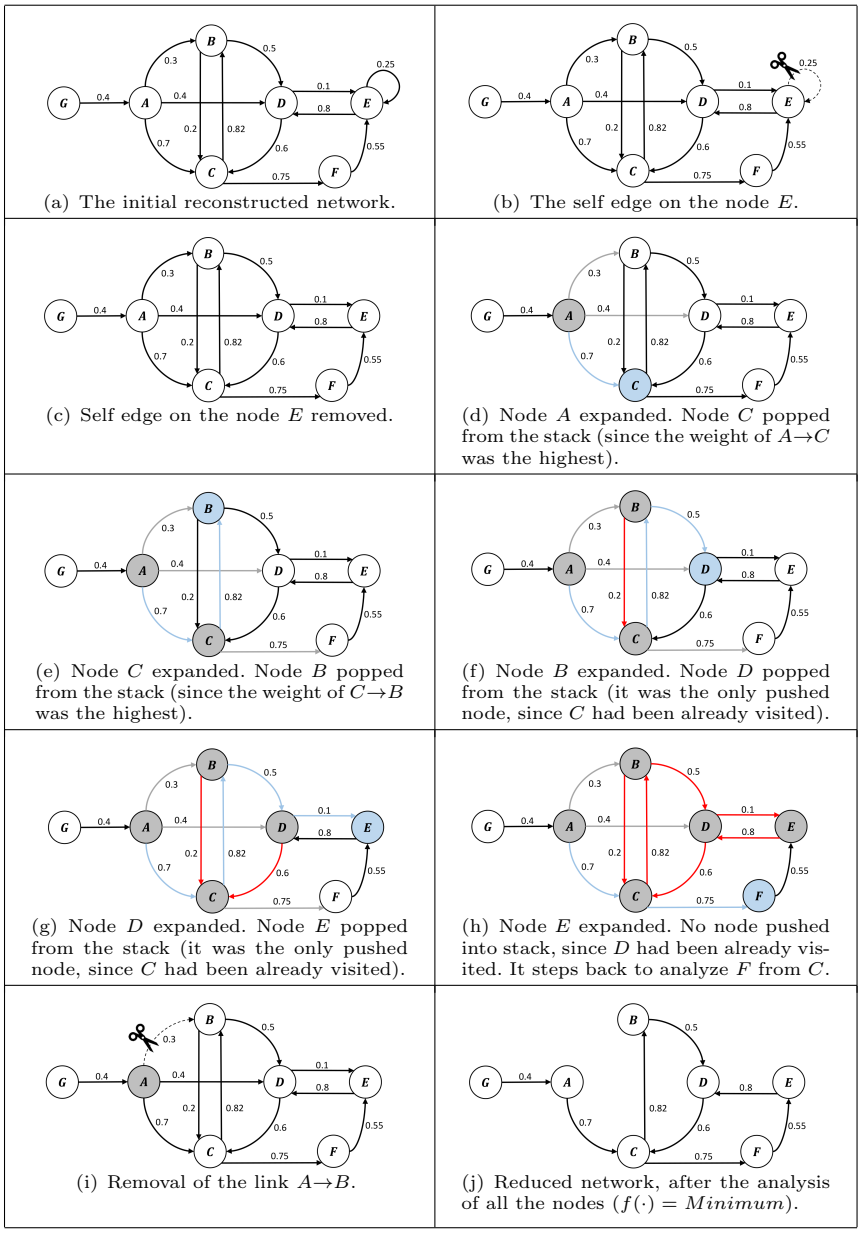
The analysis of a source node is performed as follows. First, the data structures are initialized (line 4), by considering the source node as already expanded and by adding it to the current path. Second, we analyze all its adjacent nodes, i.e., we push them into the stack, ordered in ascending order according to the edge weight, and initialize the routing table by setting themselves as their next-hop (lines 5-7). Then, the main part of the algorithm (lines 8-28) iterates until the stack still has some nodes to analyze. In particular, LOCANDA pops a node (*current_node*) from the stack (see Figure 3(d)), marks it as visited (lines 9-10), and computes the score associated with the current path to reach the *current_node* from the source (lines 11-13). If the current path is the first identified path to reach *current_node* or it has a higher score with respect to the previous path in the routing table, LOCANDA updates the routing table (lines 14-16).

Then the algorithm expands the current node, by pushing its adjacent nodes into the stack in ascending order with respect to the edge weight, if not already visited (lines 17-19). If at least a node was pushed (see Figures 3(e), 3(f), 3(g)), the current path is updated to follow *current_node* (lines 20-22), otherwise (see Figure 3(h)) the algorithm steps back, until it can find an existing edge between the last node in the path and the next node in the stack (lines 23-28). In both cases, the path and its score are updated incrementally (lines 22 and 26-28).

When there is no more nodes in the stack, LOCANDA removes all the direct links such that the properties described before are satisfied. In particular, it removes a link between the source node u and an its adjacent v if v is never used as next-hop to reach other nodes and if the path identified to reach v from u appears more reliable then the direct link (lines 29-32). The algorithm then proceeds with the next source node. It is noteworthy that the removed links will never be considered again from the algorithm. This can be done without any risk to lose relevant paths, since those edges would never be considered in any case, even analyzing the nodes of the networks in a different order. As an example, the removed edge between A and B in Figure 3(i) would not be followed in any case during the analysis of the node G as source node. Therefore, the order of analysis of source nodes does not affect the resulting reduced network.

The immediate removal of such links also improves the algorithm time complexity. Indeed, although in the pessimistic case LOCANDA has a time complexity of $O(|V| \cdot |E|)$[1], this choice decreases the number of edges at each iteration.

[1] For space constraint, we do not prove formally the time complexity of the algorithm.



(a) The initial reconstructed network.

(b) The self edge on the node E.

(c) Self edge on the node E removed.

(d) Node A expanded. Node C popped from the stack (since the weight of $A \rightarrow C$ was the highest).

(e) Node C expanded. Node B popped from the stack (since the weight of $C \rightarrow B$ was the highest).

(f) Node B expanded. Node D popped from the stack (it was the only pushed node, since C had been already visited).

(g) Node D expanded. Node E popped from the stack (it was the only pushed node, since C had been already visited).

(h) Node E expanded. No node pushed into stack, since D had been already visited. It steps back to analyze F from C.

(i) Removal of the link $A \rightarrow B$.

(j) Reduced network, after the analysis of all the nodes ($f(\cdot) = Minimum$).

Fig. 3. An example of execution of LOCANDA and the analysis of the *source node* A. Grey nodes: already expanded; blue node: the current node to analyze, extracted from the stack; black edges: not seen yet; grey edges: already seen, but still not followed; blue edges: belonging to the current path; red edges: will not be followed, since would bring to already expanded nodes; black-dashed edges: to be removed.

4 Experiments

We performed our experiments on the datasets considered in [4]. These datasets consist of steady-state expression data (10 conditions), generated by the tool SynTReN [16] on the basis of the well-defined regulatory networks of the organisms *E. coli* and *S. cerevisiae* (henceforth Yeast) [6]. SynTReN selects connected sub-networks of the input networks and generates gene expression data which best describe the network structure. We consider sub-networks of 100 and 200 genes, characterized by 121 and 303 links, with an average node degree of 2.42 and 3.03, respectively. In order to evaluate the robustness to noise, coherently to [4], we consider three versions of each dataset, with different levels of (additive, lognormally-distributed) noise, i.e., 0.0 (without noise), 0.1 and 0.5, introduced by SynTReN. Gene regulatory networks were reconstructed by adopting the system GENERE [4], which, according to the experiments, obtains state-of-the-art results in terms of Area Under the ROC Curve (AUC). In particular, we selected the parameter configuration of GENERE which led to the best results.

We considered as a competitor the system ARACNE [11], that we already described in Section 2. Moreover, we considered, as a baseline, the original network reconstructed by GENERE. For all the systems, we performed the experiments by imposing a lower threshold on the weight of the edges in $\{0.0, 0.1, \ldots 1.0\}$. For LOCANDA, we performed the experiments with all the measures for the estimation of the reliability of the path proposed in Section 3, that are: minimum (Min), product (Prod), average (Avg) and weighted average (WAvg).

The evaluation measure that we consider is based on the Area Under the ROC Curve. It is noteworthy that the classical AUC evaluation focuses on known examples in the gold standard, disregarding all the predicted links for which the existence is unknown. This means that the obtained AUC value can be significantly distorted, since focused only on the small subset of known links in the reconstructed network. Since, in real scenarios, the biologists have to analyze the whole set of predicted links, possibly ranked in descending order with respect to their score, we define the weighted AUC as follows:

$$WAUC(V, \widetilde{E}) = \left(1 - \frac{sumOfWeights(\widetilde{E})}{|V| \cdot (|V| - 1)}\right) \cdot AUC(\widetilde{E}) \tag{1}$$

where $sumOfWeights(\widetilde{E}) = \sum_{\langle u,v,w\rangle \in \widetilde{E}}(w)$ is the sum of edge weights in the reduced reconstructed network, $AUC(\widetilde{E})$ is the classical Area Under the ROC Curve and $|V| \cdot (|V| - 1)$ is the number of possible links in the network. It is noteworthy that this measure penalizes the original AUC score proportionally to the number (and the weight) of links in the reduced network. This is motivated by the fact that a large set of predicted links, all with a high score (i.e., without a clear indication about their rank) would require an extensive manual analysis performed by biologists. On the other hand, reconstructed networks with many links will not be penalized significantly if a large set of links has a very low score, since they would be probably disregarded by biologists during their analysis.

Note that, due to the weighting defined in Equation 1, WAUC values near to 0.5 do not correspond to a random prediction as in the standard AUC evaluation.

The obtained results are plotted in the box plots depicted in Figure 4. Box plots are drawn by considering the different values for the input threshold on the edge weight. This allows us to evaluate the stability of the results with respect to such a parameter. First, we can observe that ARACNE, GENERE and LOCANDA_Prod obtain unstable results with respect to the input threshold, whereas the other variants of LOCANDA obtain very stable results. Moreover, we can observe that the networks reconstructed by GENERE appear, in general, accurate and often lead to the highest WAUC value (see the datasets Ecoli-100-0.1, Ecoli-200-0.1, Ecoli-200-0.5, Yeast-100-0.1, Yeast-100-0.5, Yeast-200-0.0 and Yeast-200-0.1). However, such a result can be obtained with a specific value of the input threshold and a wrong decision can lead to very poor results. On the contrary, a non-optimal choice of the value for the input threshold does not affect significantly the results obtained by LOCANDA_Min, LOCANDA_Avg and LOCANDA_WAvg, that lead to stable and high WAUC values in almost all the cases. ARACNE generally obtains lower WAUC values, which also appear highly dependent on the value of the input threshold. An exception can be observed in the dataset Ecoli-200-0.0 in which ARACNE obtains the best result.

Analyzing the influence on the results caused by the presence of noise in the data, we can observe that, without noise or with a low amount of noise, GENERE and ARACNE obtain acceptable (although unstable) results. However, when the amount of noise increases, their average WAUC values decrease significantly. On the contrary, LOCANDA, especially with the variants based on Min, Avg and WAvg, generally shows good and stable results, even in the case of the datasets with the highest noise. This proves that the proposed method is actually very robust to the possible presence of noise in the data.

Finally, we performed the Friedman test with the Nemenyi post-hoc test, with $\alpha = 0.05$, in order to evaluate whether the obtained results appear significant from a statistical viewpoint. Following [5], we plot a graph which summarizes the results in Figure 5. Observing the graph, we can conclude that, although LOCANDA_Min generally leads to the best results, the difference among the three variants based on Min, Avg and WAvg is not statistically significant. However, the difference between the results obtained by these three variants and by the other approaches, including ARACNE and GENERE, is statistically significant.

The non-optimal results obtained by the variant based on product can be motivated by the fact that it is based on assumptions that are often violated in biological networks (i.e., the independence between the events). On the contrary, the very good results obtained by the variants Min and WAvg can be motivated by the fact that their assumptions correctly reflect the real interactions among genes. At this respect, we can conclude that: *i)* the variants based on Min and WAvg are the most appropriate for the reconstruction of gene networks, and *ii)* LOCANDA can be easily adapted to analyze networks representing data about other application domains, by identifying a proper function $f(\cdot)$ able to catch specific assumptions of the domain at hand.

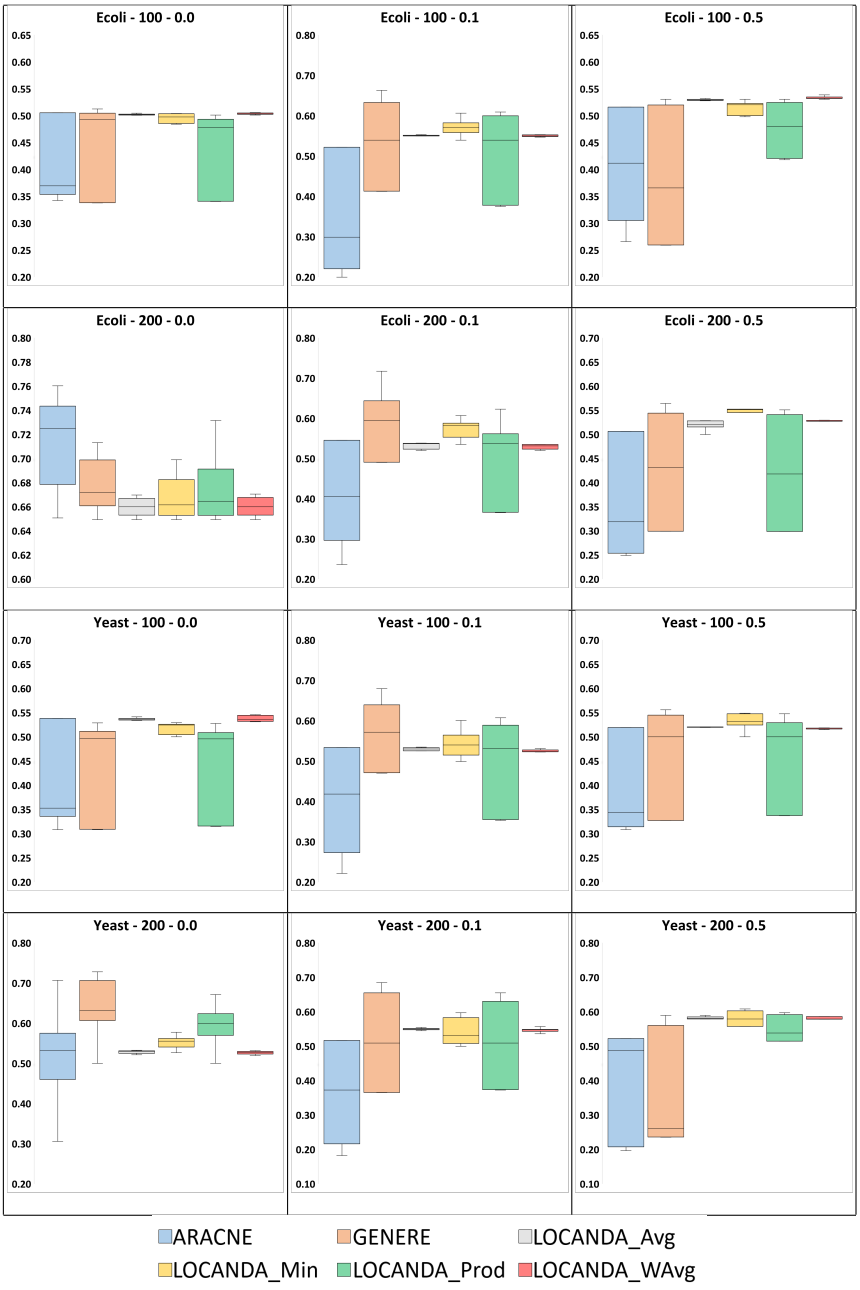


Fig. 4. Box plots depicting the results. On the X-axis there are the different methods; on the Y-axis there is the WAUC obtained by varying the threshold on the edge weight.

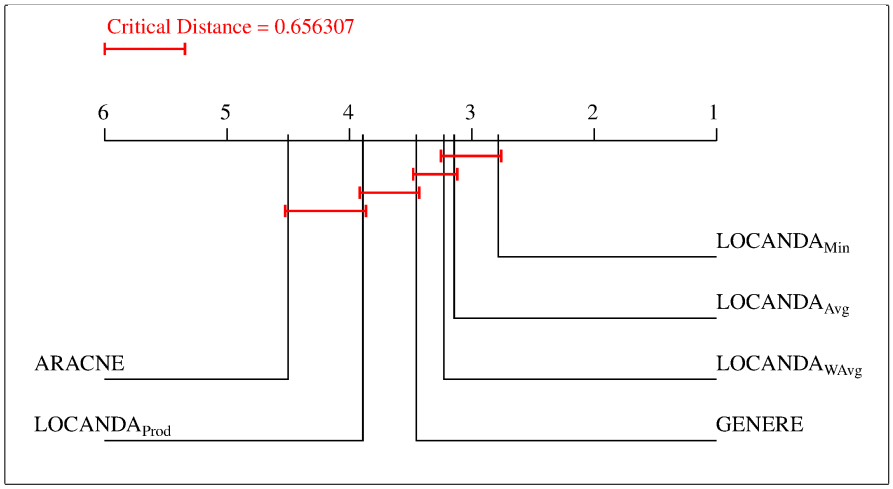


Fig. 5. Results of the Friedman test and Nemenyi post-hoc test with $\alpha = 0.05$.

5 Conclusions and Future Work

In this work, we proposed the method LOCANDA for the analysis of reconstructed biological networks, which identifies and exploits causality phenomena to remove links which can be considered the result of indirect regulation activities. Contrary to existing methods for the identification of a transitive reduction of a network or for the identification of redundancies in reconstructed biological networks, LOCANDA simultaneously offers all the following characteristics: *i)* it is able to analyze directed weighted networks, fully exploiting the weights on the edges which represent their reliability; *ii)* it does not require any pre-processing step on the network in order to handle the possible presence of cycles; *iii)* it is able to identify indirect relationships of arbitrary length and to exploit them to remove direct links considered as false positives. The estimation of the reliability of a path is guided by a function, which can be tuned according to specific underlying phenomena and assumptions with respect to the application domain at hand. Focusing on biological networks, the obtained results show that LOCANDA, especially in its variant based on minimum, is able to obtain better and more stable results with respect to the considered competitors, even with highly noisy data. Moreover, according to the Friedman test and Nemenyi post-hoc test, such difference appears statistically significant.

As future works, we plan to compare LOCANDA with additional competitor systems, also in the analysis of a larger network about the Homo Sapiens. We will also perform a qualitative analysis of the results, guided by experts in biology. Moreover, we will evaluate the effectiveness of LOCANDA in the analysis of networks representing data about other domains, focusing on the influence of the function $f(\cdot)$ when different assumptions on the network are verified.

Acknowledgements

We would like to acknowledge the support of the European Commission through the projects MAESTRA - Learning from Massive, Incompletely annotated, and Structured Data (Grant Number ICT-2013-612944) and TOREADOR - Trustworthy Model-aware Analytics Data Platform (Grant Number H2020-688797).

References

1. A. V. Aho, M. R. Garey, and J. D. Ullman. The transitive reduction of a directed graph. *SIAM J. Comput.*, 1(2):131–137, 1972.
2. N. Atias and R. Sharan. Comparative analysis of protein networks: hard problems, practical solutions. *Communications of the ACM*, 55(5):88–97, 2012.
3. D. Bošnački, M. R. Odenbrett, A. Wijs, W. Ligtenberg, and P. Hilbers. Efficient reconstruction of biological networks via transitive reduction on general purpose graphics processors. *BMC Bioinformatics*, 13(1):281, 2012.
4. M. Ceci, G. Pio, V. Kuzmanovski, and S. Džeroski. Semi-supervised multi-view learning for gene network reconstruction. *PLOS ONE*, 10(12):1–27, 12 2015.
5. J. Demšar. Statistical comparisons of classifiers over multiple data sets. *J. Mach. Learn. Res.*, 7:1–30, Dec. 2006.
6. S. Hempel, A. Koseska, Z. Nikoloski, and J. Kurths. Unraveling gene regulatory networks from time-resolved gene expression data - A measures comparison study. *BMC Bioinformatics*, 12(1):292, 2011.
7. H. T. Hsu. An algorithm for finding a minimal equivalent graph of a digraph. *J. ACM*, 22(1):11–16, Jan. 1975.
8. S. Itani, M. Ohannessian, K. Sachs, G. P. Nolan, and M. A. Dahleh. Structure learning in causal cyclic networks. In *Proc. of the Int. Conf. on Causality: Objectives and Assessment - Vol. 6*, COA'08, pages 165–176. JMLR.org, 2008.
9. K. B. Korb and A. E. Nicholson. *Bayesian Artificial Intelligence, Second Edition*. CRC Press, Inc., Boca Raton, FL, USA, 2nd edition, 2010.
10. L. Lo, M. Wong, K. Lee, and K. Leung. Time delayed causal gene regulatory network inference with hidden common causes. *PLOS ONE*, 10(9):1–47, 09 2015.
11. A. Margolin, I. Nemenman, K. Basso, C. Wiggins, G. Stolovitzky, R. Favera, and A. Califano. Aracne: An algorithm for the reconstruction of gene regulatory networks in a mammalian cellular context. *BMC Bioinformatics*, 7(Suppl 1):S7, 2006.
12. J. Pearl. *Causality: Models, Reasoning, and Inference*. Cambridge University Press, New York, NY, USA, 2000.
13. C. A. Penfold and D. L. Wild. How to infer gene networks from expression profiles, revisited. *Interface Focus*, 1(6):857–870, 2011.
14. A. Pinna, N. Soranzo, and A. de la Fuente. From knockouts to networks: establishing direct cause-effect relationships through graph analysis. *PLoS ONE*, 10(5):e12912, 2010.
15. G. Pio, M. Ceci, D. Malerba, and D. D'Elia. ComiRNet: a web-based system for the analysis of miRNA-gene regulatory networks. *BMC Bioinformatics*, 16(9):S7, 2015.
16. T. Van den Bulcke, K. Van Leemput, B. Naudts, P. van Remortel, H. Ma, A. Verschoren, B. De Moor, and K. Marchal. SynTReN: A generator of synthetic gene expression data for design and analysis of structure learning algorithms. *BMC Bioinformatics*, 7:43, 2006.
17. M. Zitnik and B. Zupan. Data Imputation in Epistatic MAPs by Network-Guided Matrix Completion. *Journal of Computational Biology*, 22(6):595–608, 2015.