Modeling parsimonious putative regulatory networks: complexity and heuristic approach

Vicente Acuña, Andrés Aravena, Alejandro Maass, Anne Siegel

Modeling parsimonious putative regulatory networks: complexity and heuristic approach

Vicente Acuña[1,2], Andrés Aravena[1,2,5], Alejandro Maass[1,2,3], and Anne Siegel[4,5]

[1] Center for Mathematical Modeling (UMI-CNRS 2807), University of Chile, Chile
[2] Center for Genome Regulation, University of Chile, Chile
[3] Department of Mathematical Engineering, University of Chile, Chile
[4] CNRS UMR 6074, IRISA Project Dyliss, Université de Rennes 1 (UMR 6074), France
[5] INRIA, Centre Rennes-Bretagne-Atlantique, Project Dyliss, Campus de Beaulieu, Rennes, France

Abstract. A relevant problem in systems biology is the description of the regulatory interactions between genes. It is observed that pairs of genes have significant correlation through several experimental conditions. The question is to find causal relationships that can explain this experimental evidence.
A putative regulatory network can be represented by an oriented weighted graph, where vertices represent genes, arcs represent predicted regulatory interactions and the arc weights represent the p-value of the prediction. Given such graph, and experimental evidence of correlation between pairs of vertices, we propose an abstraction and a method to enumerate all parsimonious subgraphs that assign causality relationships compatible with the experimental evidence.
When the problem is modeled as the minimization of a global weight function, we show that the enumeration of scenarios is a hard problem. As an heuristic, we model the problem as a set of independent minimization problems, each solvable in polynomial time, which can be combined to explore a relevant subset of the solution space. We present a logic-programming formalization of the model implemented using Answer Set Programming.
We show that, when the graph follows patterns that can be found in real organisms, our heuristic finds solutions that are good approximations to the full model. We encoded these approach using Answer Set Programming, applied this to a specific case in the organism *E. coli* and compared the execution time of each approach.

Keywords: Genic regulatory network reconstruction, Complexity, Algorithm, Heuristics

Optimization methods and abstract methods such as static analysis or model-checking appear to have more and more interplays. For instance, optimization-based approaches are required in static analysis to handle the complexity of some numerical domains. The same issue holds in systems biology. In this domain,

both static analysis and model-checking approaches have been widely developed (de Jong *et al.*, 2003; Calzone *et al.*, 2006; Chabrier-Rivier *et al.*, 2004; Danos *et al.*, 2007, 2012), but they are applied to models having strong litterature-based confidence and evidences. Meanwhile, the emergence of new sequencing technologies implies that more and more models of relatively low confidence are produced with learning and reconstruction approaches (Meyer *et al.*, 2007; Herrgård *et al.*, 2004; Marbach *et al.*, 2010). A main prospective issue is now to apply model-checking approaches to such uncertain models reconstructed from "omics" data.

As a first step in this direction, in this paper, we propose to use logic programming approaches to analyze "rough" data in order to build a robust and valid biological model that can be considered as a entry for formal approaches. More precisely, we address the issue of reconstructing a minimal graph model of regulatory interaction from genome and co-expression information. We prove that the underlying combinatorial problem is of high complexity, and we introduce a less complex (also NP-complete) heuristics to solve the problem. Then, interestingly, we propose to use a quite recent paradigm of logic programming, named Answer Set Programming (Gebser *et al.*, 2011), to solve the combinatorial problem. Interestingly, it appears that the progress of solvers developed for ASP now allows solving the heuristics that we have introduced.

1 Introduction

Molecular biology is source of many interesting graph problems. For instance, the transcriptional regulation network of an organism is usually represented by a directed graph where nodes represent genes and arcs connect each regulator gene to a regulated one. In theory the knowledge of the complete regulation network, including regulations signs (activation or repression), would allow a complete description of the cell behavior as a dynamical system (Xiao, 2009).

The set of genes is called the *genotype* of the organism, and the physical outcome that includes metabolites, proteins and the cell shape, constitute the *phenotype*. So the genotype is the potential outcome of a cell, while the phenotype is the effective outcome. The regulation network encodes the mechanism that enables a fixed genotype to become different phenotypes, for example when a multicellular organism develops and tissues are formed.

If we describe metaphorically a cell as a mechanical clock, the genetic information is the blueprint that describes each one of the gears. Genetic network reconstruction methods aim to describe how these gears are interconnected and how they interact for a given outcome. The long term goal is to describe accurately these interactions in a way that allow us to predict the effect of a change in the mechanism and, in principle, determine which modifications have to be made to obtain a desired result.

The regulation network can be modeled as a set of coupled differential equations, where each node is represented by a variable and the arcs represent the subset of these variables that are relevant for each equation. Unfortunately, these

equations usually depend on parameters that are not easily measured. Another approach is to model them as boolean networks, where the nodes can be "active" or "inactive" and the next state of each node depends only on the state of the nodes connected by incoming arcs.

The purpose of gene regulation network reconstruction has thus at least two aspects. It constitutes new scientific knowledge as it describes the basis of the behavior of a cell. It is also the basis for biotechnological applications, like new antibiotics or genetic engineering (Davidson and Levin, 2005).

These networks are hard to determine experimentally (Streit *et al.*, 2013). Instead, many approximative methods build putative networks using pattern matching techniques in the DNA sequence (Bailey *et al.*, 2009). These methods have usually low specificity (Medina-Rivera *et al.*, 2011) and the resulting putative networks have a number of arcs around ten times larger than the expected. Putative networks are represented by weighted digraphs where each gene is represented by a vertex, an arc connects two vertices when the pattern matching suggests that the first gene regulates the second, and the arc weight is related to the pattern matching score or p-value.

Another approach to understand the genetic regulation is to observe the behavior of all genes through several conditions and determine correlations among genes that suggest that some of them are (indirectly) controlled by the same regulator. This is usually done using differential expression data from microarrays experiments. When the expression level of a gene is not independent from the level of another gene, they are called a pair of co-expressed genes. This association can be measured using linear correlation or mutual information, among other techniques (Butte and Kohane, 2000).

We can integrate these two kinds of data and use the experimental evidence as a constraint to define valid putative arc predictions and to determine parsimonious graphs that represent the transcriptional regulatory network. We are interested in the enumeration of all subgraphs that satisfy a connectivity restriction and are minimal in some sense. Our approach (Aravena *et al.*, 2013) to this parsimonious description is to find the subgraphs representing networks that have, for each pair of vertices representing co-regulated genes, at least one vertex that precedes them, directly or indirectly.

The arcs of the graph resulting from our method are novel targets for experimental validation. One of the advantages of our method over the classical tools is that the average degree is reduced, thus this validations can be focused on a few cases, decreasing experimental time and cost. Once some arc have been validated, this new knowledge can be easily incorporated into our model and close the loop between experiments and modeling. This iterative process alternating theoretical analysis and practical validation is classic in systems biology.

The paper is organized as follows: the next two sections explore parsimony by enumerating minimal subgraphs considering two definitions of graph minimality, the complexity of these problems is determined, in Section 4 an heuristic approach is described, finally in Section 5 this heuristic is applied to a well known organism.

2 Arc minimal subgraphs

In the following, $\mathcal{V}$ represents the set of all genes and $\mathcal{A}_0$ represents all putative regulatory relationships. We also have a collection $\mathcal{O} \subseteq \mathcal{P}_2(\mathcal{V})$ whose elements are subsets of $\mathcal{V}$ with cardinality 2, that is, unordered pairs $\{t, t'\}$ of distinct vertices (i.e. $t \neq t'$). This collection represents the pairs of co-regulated genes.

In order to obtain parsimonious regulatory graphs we need to compute subgraphs with a minimal set of arcs that can explain all experimental evidence. Thus, the solutions to our problem are completely defined by their set of arcs $A \subseteq \mathcal{A}_0$.

Let $\mathcal{G} = (\mathcal{V}, \mathcal{A}_0)$ be a directed graph on vertex set $\mathcal{V}$ and arc set $\mathcal{A}_0$. A graph $G = (\mathcal{V}, A)$ is a **subgraph** of $\mathcal{G} = (\mathcal{V}, \mathcal{A}_0)$, if $A \subseteq \mathcal{A}_0$.

Now, we model the condition that for each pair of co-regulated genes our subgraph should contain a common regulator.

Definition 1. *Given an arc set $A \subseteq \mathcal{A}_0$ we say that a vertex $s \in \mathcal{V}$ **precedes** a vertex $t \in \mathcal{V}$ in A if there exists an oriented path from s to t using only arcs in A. In particular every node $v \in \mathcal{V}$ precedes itself.*

Definition 2. *We say that an arc set A is **$\mathcal{O}$-coherent** if each pair in $\mathcal{O}$ satisfies the **precedence condition**:*

$$\forall \{t, t'\} \in \mathcal{O} \quad \exists s \in \mathcal{V}, \quad s \text{ precedes } t \text{ in } A \wedge s \text{ precedes } t' \text{ in } A.$$

We also say that the subgraph $G = (\mathcal{V}, A)$ is $\mathcal{O}$-coherent when its arc set A is $\mathcal{O}$-coherent.

We assume that $\mathcal{A}_0$ is $\mathcal{O}$-coherent. Notice that, for each $\{t, t'\} \in \mathcal{O}$, if A contains a directed path from t to t' then the precedence condition is automatically satisfied by choosing $s = t$.

The idea is to describe the subsets of $\mathcal{A}_0$ which are $\mathcal{O}$-coherent. Notice that the property of being $\mathcal{O}$-coherent is monotone: if A is $\mathcal{O}$-coherent then every graph containing A is also $\mathcal{O}$-coherent. Thus, we are interested in enumerate only the subgraphs that are *minimal* in the following sense:

Definition 3. *We say that an $\mathcal{O}$-coherent arc set A is **minimal $\mathcal{O}$-coherent** if for any $a \in A$ we have that $A - a$ is not $\mathcal{O}$-coherent. We say that the subgraph $G = (\mathcal{V}, A)$ is minimal $\mathcal{O}$-coherent when its arc set A is minimal $\mathcal{O}$-coherent.*

Checking if a subgraph G is $\mathcal{O}$-coherent can be done in polynomial time. For each $\{t, t'\} \in \mathcal{O}$ we build the sets of all predecessors of t and all predecessors of t' in linear time. If the intersection is not empty for all pair $\{t, t'\} \in \mathcal{O}$ then G is $\mathcal{O}$-coherent. Therefore, it is easy to find *one* minimal $\mathcal{O}$-coherent subgraph of $\mathcal{G}$. By iteratively removing arcs of $\mathcal{G}$ while the condition is maintained we obtain a minimal graph in quadratic time. Consider the following problem:

> EnumCohe$(\mathcal{G}, \mathcal{O})$: Given an oriented graph $\mathcal{G}$ and a set of pairs of vertices $\mathcal{O} \subset \mathcal{P}_2(V)$, enumerate all minimal $\mathcal{O}$-coherent subgraphs of $\mathcal{G}$.

We want to analyse the computational complexity of this enumeration problem. Notice that the number of minimal $\mathcal{O}$-coherent subgraphs of $\mathcal{G}$ can grow exponentially (consider, for instance, $\mathcal{A}_0$ a complete graph and $\mathcal{O}$ containing only one pair of vertices). Therefore, just printing the result would take exponential time in terms of the input size. In these cases, it is more appropriate to use *total time* to analyse the complexity of enumeration. That is, the time is measured in terms of the size of the input *and* the number of solutions (Johnson *et al.*, 1988). Thus, we say that EnumCohe can be done in *polynomial total time* if we can enumerate the solutions in polynomial time in the size of $\mathcal{G}$, $\mathcal{O}$ and the number of minimal $\mathcal{O}$-coherent subgraphs of $\mathcal{G}$.

Unfortunately, the problem EnumCohe is hard in following sense: enumerate all minimal $\mathcal{O}$-coherent subgraphs cannot be done in polynomial total time unless $P = NP$. To prove this, we reduce EnumCohe to the **path conjunction problem**:

> PathConj$(\mathcal{G}, \mathcal{P})$: Given an oriented graph $\mathcal{G} = (\mathcal{V}, \mathcal{A}_0)$ and a set of pairs of vertices $\mathcal{M} = \{(s_i, t_i), i = 1 \dots n\} \subseteq \mathcal{V} \times \mathcal{V}$, enumerate all minimal subsets $A \subseteq \mathcal{A}_0$ such that for each $(s_i, t_i) \in \mathcal{M}$, there is an oriented path from s_i to t_i.

Here minimality is in the subset sense: if A is minimal then it connects all pairs in $\mathcal{M}$ and for each $a \in A$ there is at least one pair in $\mathcal{M}$ that is not connected in $A - a$. Khachiyan *et al.* (2007) shows that PathConj cannot be enumerated in polynomial total time unless $P = NP$.

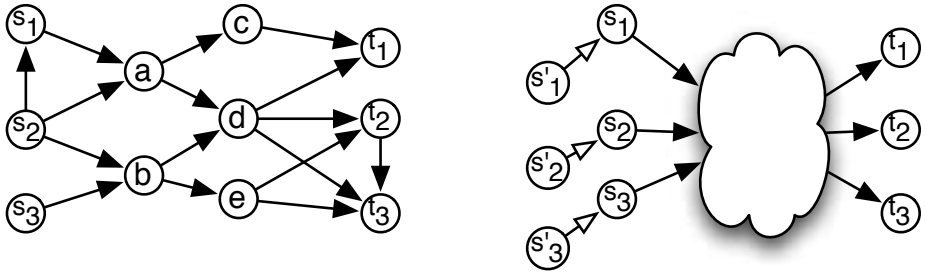

Fig. 1. (A) Example of the path conjugation problem, which enumerates all minimal subgraphs connecting pairs of vertices in $\mathcal{M} = \{(s_1, t_1), (s_2, t_2), (s_3, t_3)\}$. One such subgraph is the induced by the vertices a, b and d. **(B) Reduction of the path conjugation problem to EnumCohe**. Additions of the s'_i nodes guarantees that each s_i is connected to the corresponding t_i, as described in the text. The latter problem is thus as complex as the first.

Theorem 1. *Problem* EnumCohe *cannot be solved in polynomial total time unless* $P = NP$.

Proof. Problem PathConj can be reduced to EnumCohe in linear time. Let us consider $\mathcal{G} = (V, \mathcal{A}_0)$ and $\mathcal{M} = \{(s_i, t_i), i = 1 \dots n\}$ an instance of PathConj. We can create an instance for EnumCohe to solve this problem. Define the graph $G' = (V \cup V', \mathcal{A}_0 \cup \mathcal{A}_0{}')$ where $V' = \{s'_i, i = 1 \dots n\}$ and $\mathcal{A}_0{}' = \{(s'_i, s_i), i = 1 \dots n\}$. Consider the set of pairs $\mathcal{O} = \{(s'_i, t_i), i = 1 \dots n\}$. Clearly each minimal $\mathcal{O}$-coherent subgraph of G' is exactly the set of arcs in A' union a minimal subgraph connecting the pairs in $\mathcal{M}$. Then, there is a one-to-one correspondence between the solutions of EnumCohe$(G', \mathcal{O})$ and the solutions of PathConj$(\mathcal{G}, \mathcal{P})$.

3 Minimum weight subgraphs

The graphs that represent putative regulatory networks are built using pattern matching techniques that determine when a given gene can be a regulator (Altschul *et al.*, 1997) and which genes can it regulate (Bailey *et al.*, 2009) based on the DNA sequence of the genome. This prediction is characterized by the score of each gene versus the reference pattern, and by a p-value that states the probability of observing that score under the null hypothesis that there not exists a regulation relationship. A lower p-value corresponds to a higher confidence that the arc corresponds to a real regulatory relationships.

We assume that each arc in $\mathcal{A}_0$ has a positive weight that increases with the p-value of the arc. Then each subgraph has a global weight, and a parsimonious regulatory graph is any $\mathcal{O}$-coherent subgraph of minimum weight.

Let $w : \mathcal{A}_0 \to \mathbb{N}$ be the function that assigns a non-negative weight to each arc in $\mathcal{A}_0$. Then the weight (or cost) of an arc-set A is $W(A) = \sum_{a \in A} w(a)$. We are interested in finding a $\mathcal{O}$-coherent subgraph of minimum weight. It is easy to see that any minimum weight $\mathcal{O}$-coherent subgraph is also arc minimal, but not all arc minimal subsets have minimum weight. Unfortunately, even finding one $\mathcal{O}$-coherent subgraph of minimum weight is *NP*-hard. We define formally this problem as MinCohe:

> MinCohe$(\mathcal{G}, \mathcal{O})$: Given an oriented graph $\mathcal{G}$ and a set of pairs of vertices $\mathcal{O} \subset \mathcal{P}_2(\mathcal{V})$, find a $\mathcal{O}$-coherent subgraph of minimum weight.

To prove MinCohe is *NP*-hard, we introduce the Steiner Weighted Directed Tree problem:

> SWDT$(\mathcal{G}, s, T)$: Given an oriented weighted graph $\mathcal{G} = (\mathcal{V}, \mathcal{A}_0)$, a vertex $s \in \mathcal{V}$ and a set of vertices $T = \{t_i, i = 1 \dots n\} \subseteq \mathcal{V}$, find a subgraph of minimum weight that connect s to t_i for all $t_i \in T$.

The problem SWDT is *NP*-hard. Indeed, the undirected case of this problem corresponds, in their decision version, to one of Karp's 21 *NP*-complete problems (Karp, 1972). Since SWDT is an extension of the undirected case, it is also *NP*-hard.

Theorem 2. *Problem* MinCohe *is NP-hard.*

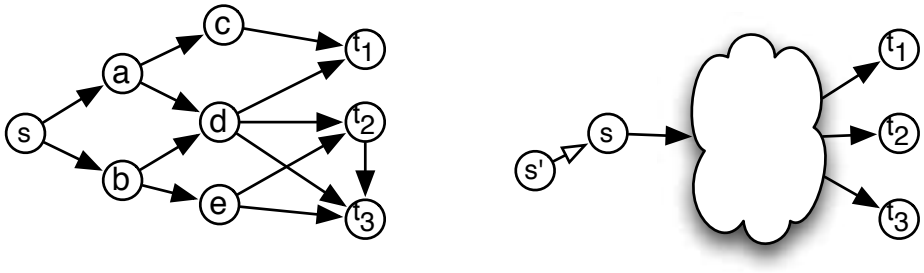

Fig. 2. (A) Schema of Steiner Directed Weighted Tree (SDWT), which enumerates all minimum weight subgraphs connecting s to vertices in $T = \{t_1, t_2, t_3\}$. For example the tree induced by nodes a and d connects s with T with minumum weight. **(B) Reduction of Steiner Directed Weighted Tree problem to MinCohe**. The latter problem is thus as complex as the first one.

Proof. We reduce SWDT problem to MinCohe in a similar way than in the previous result. Let us consider $\mathcal{G} = (\mathcal{V}, \mathcal{A}_0)$, $s \in V$ and $T = \{t_i, i = 1 \dots n\}$ an instance of SWDT. Define the graph $G' = (\mathcal{V} \cup \{s'\}, \mathcal{A}_0 \cup \{(s', s)\})$ where s' is a new vertex and (s', s) is a new arc with weight zero. Consider the set of pairs $\mathcal{O} = \{(s', t_i), i = 1 \dots n\}$. Clearly a solution of MinCohe$(G', \mathcal{O})$ is exactly the singleton $\{(s', s)\}$ union a solution of SWDT$(\mathcal{G}, s, T)$.

4 Subgraphs with minimum weight paths

We define a *v-shape* as the union of two directed paths starting from the same vertex with no other vertex in common. Formally,

Definition 4. *Let s, t and t' be three vertices of G with $t \neq t'$. Let P be a directed path from s to t and let P' be a directed path from s to t' such that P and P' have only vertex s in common. Then, we say that $Q = P \cup P'$ is a* **v-shape**. *We also say that vertices t and t' are* **v-connected** *by Q.*

Clearly if an arc set $A \subseteq \mathcal{A}_0$ is $\mathcal{O}$-coherent, then for each pair $\{t, t'\}$ in $\mathcal{O}$ there is at least one v-shape in $G(\mathcal{V}, A)$ that v-connects t and t'. Thus, if we consider local parsimony principle, for each pair $\{t, t'\}$ in $\mathcal{O}$ we should include in our solution A a v-shape of minimum weight v-connecting t and t'.

This is not necessarily the case for the solutions given by MinCohe. Indeed, a solution G of MinCohe has minimum *global* weight, but this does not imply that every pair is v-connected by a minimum weight v-shape, as can be seen in Fig. 3.

In the following, we would like to consider only $\mathcal{O}$-coherent subgraphs that contain a minimum weight v-shape for each pair in $\mathcal{O}$. We first define the collection of all v-shapes of minimum weight connecting two vertices in our initial graph $G(\mathcal{V}, \mathcal{A}_0)$:

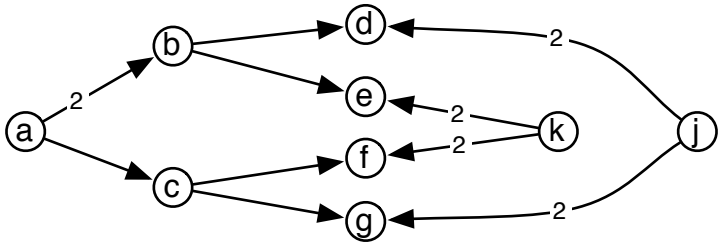

Fig. 3. Example graph where MinCohe solution is not formed by a minimum weight v-shapes. If $\mathcal{O} = \{\{d, g\}, \{e, f\}\}$ then the MinCohe solution has weight 7 and uses the arcs $(a, b), (b, d), (b, e), (a, c), (c, f), (c, g)$. An $\mathcal{O}$-short solution has weight 8. In contrast, when $\mathcal{O} = \{\{d, e\}, \{f, g\}\}$, both solutions coincide. Arcs have weight 1 unless otherwise declared.

Definition 5. *Given a graph $G(\mathcal{V}, \mathcal{A}_0)$, we call **Short-v-shape**(t, t') to the collection of all v-shapes that v-connect t and t' and are of minimum weight in $\mathcal{A}_0$.*

Now, we can define the solutions that contain a minimum weight v-shape for every pair in $\mathcal{O}$.

Definition 6. *Given a $\mathcal{O}$-coherent arc set $A \subseteq \mathcal{A}_0$, we say that A is **$\mathcal{O}$-short** if the subgraph $G(\mathcal{V}, A)$ contains a v-shape in Short-v-shape(t, t') for each pair $\{t, t'\} \in \mathcal{O}$.*

We are interested in finding the $\mathcal{O}$-coherent subgraphs that are $\mathcal{O}$-short. In particular we are interested in those $\mathcal{O}$-short having minimum weight. We propose the following problem:

> MinWeightOshort$(\mathcal{G}, \mathcal{O})$: Given an oriented graph $\mathcal{G} = (\mathcal{V}, \mathcal{A}_0)$ and a set of pairs of vertices $\mathcal{O} \subset \mathcal{P}_2(\mathcal{V})$, find a $\mathcal{O}$-short subgraph of minimum weight.

The following result is proved by a reduction from the NP-complete problem Hitting set (see Garey and Johnson, 1979): given a set of elements $A = \{1, \ldots, m\}$ and a collection of subsets $\mathcal{I} = \{I_1, \ldots, I_n\}$ of A, find a minimum cardinality subset of elements $H \subseteq A$ such that $H \bigcap I_i \neq \emptyset$, $\forall i = 1, \ldots, n$.

Theorem 3. *The problem* MinWeightOshort *is NP-hard.*

Proof. Let A and $\mathcal{I} = \{I_1, \ldots, I_n\}$ be an instance of hitting set problem. We consider the the graph $G(\mathcal{V}, A)$, where for each element a in A there are two vertices a and a' and an arc from a to a' of weight one. Additionally, for each set I_i with $i \in \{1, \ldots, n\}$ there are two vertices I_i and I_i'. Moreover, if a belongs to I_i , then there are two arcs of weight zero: one from vertex I_i to vertex a and one from vertex a' to vertex I_i'. If we define the set $\mathcal{O}$ by including all the pairs of vertices $\{I_i, I_i'\}$, then clearly any $\mathcal{O}$-short subgraph of minimum weight correspond to a minimum cardinality hitting set of the original problem.

Although this problem is theoretically hard, it could be much more tractable than the previous formulations for the instances that we are interested. Indeed, the combinatorial explosion of feasible solutions can be controlled if the size of the collections *Short-v-shape*(t, t') is small for every pair $\{t, t'\}$ in $\mathcal{O}$. That is, the number of v-shapes of minimum weight between each pair of vertices in $\mathcal{O}$ is small.

Thus, we can use a complete enumeration of unions generated by choosing one v-shape for each pair. At the end we select those unions of minimum weight.

Notice that, for a pair $\{t, t'\} \in \mathcal{O}$, computing the set *Short-v-shape*(t, t') can be done in polynomial total time by using some clever modification of the Dijkstra's algorithm (Dijkstra, 1959).

5 An illustrative example

To evaluate these approaches we consider a toy problem on a well known organism. We build a graph $\mathcal{G}$ using the genomic DNA sequence of the bacteria *E. coli* and patterns described in RegulonDB. We identified as putative regulators the genes with high homology to know genes coding for transcription factors following a standard protocol: using Blast (Altschul *et al.*, 1997) with a cutoff E-value of 10^{-10}. Then we determined where these transcription factors could bind using the tool FIMO from the MEME suite (Bailey *et al.*, 2009) to determine the presence of putative binding sites in the upstream region of every gene, with a p-value cutoff of 10^{-5}. When these binding sites were found we connected with an arc each regulator gene to the gene located downstream the binding site. This protocol is basically the classical protocol used for regulatory network reconstruction.

As described, the arcs of this graph are inferred using probabilistic tools that characterize them with a E-value and a p-value. We combined these two values and ranked all the arcs into three categories: high, medium and low confidence. We assigned discrete weights to each arc according to this classification. High confidence arcs have weight equal to 1, medium confidence arcs have weight 10 and high confidence arcs have weight 100.

We determined the set $\mathcal{O}$ of pairs of co-expressed genes from a set of 133 differential expression experiments, estimating the mutual information among them using the Pearson method and choosing the relevant relationships by the *Maximum Relevance Minimum Redundancy* (MRNET) criteria (Meyer *et al.*, 2007). This method uses mutual information as a way to determine non-linear dependencies between the expression profiles of the genes, and then determines which dependencies are significant with the following iterative procedure: for each gene a, it determines a set S_a of potentially associated genes. Initially $S_a = \varnothing$. In each iteration MRNET determines the gene b that maximizes

$$MI(a, b) - \frac{1}{|S_a|} \sum_{c \in S_a} MI(b, c)$$

The gene b that maximizes this expression with a value over a threshold is added to the set S. This expression corresponds perfectly to the idea behind MRNET. The first term of this expression focus on finding the associated genes that are of maximal relevance for a, while the second term focus on minimizing the redundancy with respect to the associated genes already in S_a. Under these conditions the pair (a, b) is in $\mathcal{O}$. We further limited $\mathcal{O}$ to the 10 000 elements with higher mutual information.

Finally, to include an aditional biological constraint and reduce the network size we contracted the graph $\mathcal{G}$ and the set $\mathcal{O}$ using the node equivalence classes defined by *operons* as predicted in ProOpDB (Taboada *et al.*, 2012). This can be done because, in bacteria and other prokaryotes, operons are set of genes that are always expressed together so their behavior is identical.

The graph $\mathcal{G}$ contains 2215 vertices and 11 584 arcs, the set $\mathcal{O}$ contains 9442 pairs of vertices. We relied on Boolean constraint processing technology for coding MinCohe$(\mathcal{G}, \mathcal{O})$, given that it is highly effective for solving demanding combinatorial problems. To be more precise, we take advantage of Answer Set Programming (ASP), a declarative problem solving approach providing a declarative framework for modeling various Knowledge Representation and Reasoning problems (Baral, 2003) combining a rich yet simple modeling language with high-performance Boolean constraint solving capacities.

The pairing of declarativeness and performance in state-of-the-art ASP solvers allows for concentrating on an actual problem, rather than how to implementing it. The basic idea of ASP is to express a problem in a logical format so that the models of its representation provide the solutions to the original problem. Problems are expressed as logic programs and the resulting models are referred to as answer sets. Although determining whether a program has a answer set is the fundamental decision problem in ASP, more reasoning modes are needed for covering the variety of reasoning problems encountered in applications. Hence, a modern ASP solver, like clasp supports several reasoning modes for assessing the multitude of answer sets, among them, regular and projective enumeration, intersection and union, and multi-criteria optimization. As well, these reasoning modes can be combined, for instance, for computing the intersection of all optimal models. This is accomplished in several steps. At first, a logic program with first-order variables is turned by efficient database techniques into a propositional logic program. This is in turn passed to a solver computing the answer sets of the resulting program by using advanced Boolean constraint technology. For optimization, a solver like clasp uses usually branch-and-bound algorithms (other choices, like computing unsatisfiable cores, exist). The enumeration of all optimal models is done in two steps. At first an optimal model is determined along with its optimum value. This computation has itself two distinct phases. First, an optimal model candidate must be found and second, it must be shown that there is no better candidate; the latter amounts to a proof of unsatisfiability and is often complex. Then, all models possessing the same value are enumerated in a second step.

Our encodings are written in the input language of gringo 3 (Gebser *et al.*, 2009, 2011). In what follows we introduce its basic syntax and we refer the reader to the corresponding literature for more details. An atom is a constant (e.g. `p, q`) or a function symbol (e.g. `p(a,b), q(X,10)`) where uppercase denotes first-order variables. Then, a rule is of the form

$$H \texttt{:-}\ B_1, \ldots, B_n.$$

where H (head) is an atom and any B_j (body) is a literal of the form A or `not` A for an atom A where the connective `not` corresponds to default negation. Further, a rule without body is a fact, whereas a rule without head is an integrity constraint. A logic program consists of a set of rules, each of which is terminated by a period. The connectives `:-` and `,` can be read as *if* and *and*, respectively. A statement starting with `not` is satisfied unless its enclosed proposition is found to be true. The semantics of a logic program is given by the *stable models semantics* (Gelfond and Lifschitz, 1988). Intuitively, the head of a rule has to be true whenever all its body literals are true. In ASP every atom needs some derivation, i.e., an atom cannot be true if there is no rule deriving it. This implies that only atoms appearing in some head can appear in answer sets, i.e. stable models. We end this quick introduction by three language constructs particularly interesting for our encoding. First, the so called choice rule of the form,

$$\{H_1, \ldots, H_m\} \texttt{:-}\ B_1, \ldots, Bn.$$

allows us to express choices over subsets of atoms. Any subset of its head atoms can be included in a stable model, provided the body literals are satisfied. Note that using a choice rule one can easily generate an exponential search space of candidate solutions. Second, a conditional literal is of the form

$$L : L1 : \cdots : Ln$$

The purpose of this language construct is to govern the instantiation of the literal L through the literals $L_1, \ldots, L_n$. In this respect, the conditional literal above can be regarded as the list of elements in the set $\{L|L_1, \ldots, L_n\}$. Finally, for solving (multi-criteria) optimization problems, ASP allows for expressing cost functions in terms of a weighted sum of elements subject to minimization and/or maximization. Such objective functions are expressed in ASP in terms of optimization statements of the form

$$\texttt{\#minimize}[L_1 = W_1@P_1, \ldots, L_N = W_N@P_N].$$

where every L_j is a literal and every W_j an integer weight. Further, P_i provides an integer priority level. Priorities allow for representing lexicographically ordered minimization objectives, greater levels being more significant than smaller ones. By default all priorities are 1.

The coding, shown in Fig 4, is straight-forward. Predicates `arc(X,Y,W)` represent the arcs in $\mathcal{A}_0$ and their weights, and predicates `coexp(X,Y)` represent the elements of $\mathcal{O}$. The optimization is carried on in two stages. First the solver

```
% Input: arc(X,Y,W) means there is an arc between X and Y with weight W
% Input: coexp(X,Y) means that {X,Y} are in O

% each arc can be used or not
{ used_arc(X,Y,W) } :- arc(X,Y,W).

% node X precedes node Y
precedes(X,Y) :- used_arc(X,Y,_).
precedes(X,Y) :- precedes(X,Z), used_arc(Z,Y,_).

% motif M is an explanation of operons A and B linked by coexpressedOp/2
v_connected(A,B) :- precedes(M,A), precedes(M,B), coexp(A,B).

% all coexpressed vertices should be v-connected
:- coexp(A,B), not v_connected(A,B).

% look for minimum global weight
#minimize [used_arc(X,Y,W)=W].
```

Fig. 4. ASP code to find a solution of MinCohe.

looks for the minimum possible global weight. Then, once this value has been determined, we look for all the answer sets that realize the minimum values. In each answer set the predicates `used_arc(X,Y,W)` indicate the arcs of a subgraph satisfying MinCohe$(\mathcal{G}, \mathcal{O})$.

Execution of this program is highly time-consuming. After a week of clock time we reached the time limit of our cluster scheduler without finding the minimum weight value.

We then proceeded to solve MinWeightOshort$(\mathcal{G}, \mathcal{O})$ using the following strategy. For each $\{t, t'\} \in \mathcal{O}$ we determine the set *Short-v-shape*(t, t') using the `get.all.shortest.paths` of the *igraph* library (Csardi and Nepusz, 2006) in the R environment (R Core Team, 2012), and assigned an unique id to each one. We coded these v-shapes using the ASP predicate `vshape(ID,T1,T2)` and the arcs that form them with the predicate `arcInVshape(ID,X,Y,W)`. In this encoding `ID` corresponds to the v-shape id, `T1,T2` correspond to $t, t' \in \mathcal{O}$, `X,Y` identify the extremes of an arc, and `W` is the weight of it.

Using these predicates, and the rules in Fig. 5, we used ASP solver *unclasp* to find the minimum weight. Execution time was 15 seconds.

A second execution was performed to find all answer sets realizing that weight. Notice that this encoding can describe the same graph as combinations of different v-shapes. We used the meta-commands `#hide`, `#show used_arc/3` and the *clasp* option `project` to collapse all answer sets with the same `used_arc/3` predicates in a single answer. This second execution took 80 minutes and resulted in a unique graph.

```
% Input: vshape(I,A,B) when v-shape I is in short-v-shapes(A,B)
% Input: arcInVshape(I,X,Y,W) when v-shape I has an arc (X, Y) w/weight W
% Input: coexp(X,Y) means that {X,Y} are in the set O

% only one v-shape is chosen for each {t,t'} in O
1{ chosen(I) : vshape(I,A,B) }1 :- coexp(A,B).

% consider the arcs that are part of the chosen v-shape
used_arc(X,Y,W) :- arcInVshape(I,X,Y,W), chosen(I).

% minimize the global weight
#minimize [used_arc(_,_,W) = W].
#hide.
#show used_arc/3.
```

Fig. 5. ASP code to find a solution of MinCohe.

6 Conclusion

The proposed algorithm can enumerate MinWeightOshort solutions in practical time, providing a way to explore a relevant subset of the $\mathcal{O}$-coherent subgraphs significantly faster than solving MinCohe. In many cases, when the graph represents a real regulatory network, it is reasonable to expect that many co-expressed nodes are connected by short v-shapes. In such cases the proposed algorithm can be used as an heuristic for MinCohe.

When it is relevant to find an exact solution of MinCohe, the heuristic solution is still useful. First, it provides a good upper bound for the global weight, which can speed up the search for the optimal value. Second, a solution of MinWeightOshort is a graph that can be used as a starting point for the combinatorial exploration required by MinCohe. We think this can be applied using the new heuristic ASP solver *hclasp* in the Potassco suite.

7 Acknowledgements

We acknowledge the support by Fondap [15090007]; Basal Grant CMM; IntegrativeBioChile INRIA Associated Team; CIRIC-INRIA Chile line Natural Ressources; ANR Biotempo[ANR-10-BLANC-0218]; U. Européenne de Bretagne [to A.A.]; INRIA-Conicyt 2010–55 [to A.A. and A.M.]; and "Estadías de Investigación U. de Chile" [to A.A.].

Bibliography

Altschul, S. F., Madden, T. L., Schäffer, A. A., Zhang, J., Zhang, Z., Miller, W., and Lipman, D. J. (1997). Gapped blast and psi-blast: a new generation of protein database search programs. *Nucleic Acids Res*, **25**(17), 3389–402.

Aravena, A., Guziolowski, C., Ostrowski, M., Schaub, T., Eveillard, D., Maass, A., and Siegel, A. (2013). Deciphering regulatory relationships with a logic-based model of causality for gene expression associations. In preparation.

Bailey, T. L., Boden, M., Buske, F. A., Frith, M., Grant, C. E., Clementi, L., Ren, J., Li, W. W., and Noble, W. S. (2009). Meme suite: tools for motif discovery and searching. *Nucleic Acids Research*, **37**(Web Server issue), W202.

Baral, C. (2003). *Knowledge representation, reasoning and declarative problem solving*. Cambridge university press.

Butte, A. J. and Kohane, I. S. (2000). Mutual information relevance networks: functional genomic clustering using pairwise entropy measurements. *Pac Symp Biocomput*, pages 418–29.

Calzone, L., Fages, F., and Soliman, S. (2006). Biocham: an environment for modeling biological systems and formalizing experimental knowledge. *Bioinformatics*.

Chabrier-Rivier, N., Chiaverini, M., Danos, V., Fages, F., and Schächter, V. (2004). Modeling and querying biomolecular interaction networks. *Theoretical Computer Science*, **325**(1), 25–44.

Csardi, G. and Nepusz, T. (2006). The igraph software package for complex network research. *InterJournal*, **Complex Systems**, 1695.

Danos, V., Feret, J., Fontana, W., Harmer, R., Krivine, J., Biosystems, P., École Normale Supérieure, and École Polytechnique (2007). Rule-based modelling of cellular signalling. In *Proceedings of the 18 th International Conference on Concurrency Theory (CONCUR'07), Lecture Notes in Computer Science*, pages 17–41.

Danos, V., Feret, J., Fontana, W., Harmer, R., Hayman, J., Krivine, J., Thompson-Walsh, C. D., and Winskel, G. (2012). Graphs, rewriting and pathway reconstruction for rule-based models. In D. D'Souza, T. Kavitha, and J. Radhakrishnan, editors, *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2012*, volume 18 of *LIPIcs*, pages 276–288. Schloss Dagstuhl - Leibniz-Zentrum für Informatik.

Davidson, E. and Levin, M. (2005). Gene regulatory networks. *Proceedings of the National Academy of Sciences of the United States of America*, **102**(14), 4935.

de Jong, H., Geiselmann, J., Hernandez, C., and Page, M. (2003). Genetic network analyzer: qualitative simulation of genetic regulatory networks. *Bioinformatics*, **19**(3), 336–344.

Dijkstra, E. (1959). A note on two problems in connexion with graphs. *Numerische Mathematik*, **1**(1), 269–271.

Garey, M. R. and Johnson, D. S. (1979). *Computers and Intractability (A guide to the theory of NP-completeness)*. W.H. Freeman and Company, New York.

Gebser, M., Kaminski, R., Ostrowski, M., Schaub, T., and Thiele, S. (2009). On the input language of asp grounder gringo. In *Logic Programming and Nonmonotonic Reasoning*, pages 502–508. Springer.

Gebser, M., Kaufmann, B., Kaminski, R., Ostrowski, M., Schaub, T., and Schneider, M. (2011). Potassco: The potsdam answer set solving collection. *AI Communications*, **24**(2), 107–124.

Gelfond, M. and Lifschitz, V. (1988). The stable model semantics for logic programming. In *ICLP/SLP*, volume 88, pages 1070–1080.

Herrgård, M. J., Covert, M. W., and Palsson, B. Ø. (2004). Reconstruction of microbial transcriptional regulatory networks. *Curr Opin Biotechnol*, **15**(1), 70–7.

Johnson, D., Yannakakis, M., and Papadimitriou, C. (1988). On generating all maximal independent sets. *Information Processing Letters*, **27**(3), 119–123.

Karp, R. M. (1972). Reducibility Among Combinatorial Problems. In R. E. Miller and J. W. Thatcher, editors, *Complexity of Computer Computations*, pages 85–103. Plenum Press.

Khachiyan, L., Boros, E., Elbassioni, K., Gurvich, V., and Makino, K. (2007). Enumerating disjunctions and conjunctions of paths and cuts in reliability theory. *Discrete applied mathematics*, **155**(2), 137–149.

Marbach, D., Prill, R. J., Schaffter, T., Mattiussi, C., Floreano, D., and Stolovitzky, G. (2010). Revealing strengths and weaknesses of methods for gene network inference. *Proceedings of the National Academy of Sciences*.

Medina-Rivera, A., Abreu-Goodger, C., Thomas-Chollier, M., Salgado, H., Collado-Vides, J., and van Helden, J. (2011). Theoretical and empirical quality assessment of transcription factor-binding motifs. *Nucleic Acids Research*, **39**(3), 808–24.

Meyer, P. E., Kontos, K., Lafitte, F., and Bontempi, G. (2007). Information-theoretic inference of large transcriptional regulatory networks. *EURASIP J Bioinform Syst Biol*, page 79879.

R Core Team (2012). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0.

Streit, A., Tambalo, M., Chen, J., Grocott, T., Anwar, M., Sosinsky, A., and Stern, C. D. (2013). Experimental approaches for gene regulatory network construction: The chick as a model system. *genesis*, **51**(5), 296–310.

Taboada, B., Ciria, R., Martinez-Guerrero, C. E., and Merino, E. (2012). Proopdb: Prokaryotic operon database. *Nucleic Acids Res*, **40**(Database issue), D627–31.

Xiao, Y. (2009). A tutorial on analysis and simulation of boolean gene regulatory network models. *Current genomics*, **10**(7), 511.