Discovering Candidates for Gene Network Expansion by Distributed Volunteer Computing

Francesco Asnicar
DISI, University of Trento
f.asnicar@unitn.it

Luca Erculiani
DISI, University of Trento

Francesca Galante
DISI, University of Trento

Caterina Gallo
Dmath, University of Trento

Luca Masera
DISI, University of Trento

Paolo Morettin
DISI, University of Trento

Nadir Sella
DISI, University of Trento

Stanislau Semeniuta
DISI, University of Trento

Thomas Tolio
DISI, University of Trento

Giulia Malacarne
CRI
Fondazione Edmund Mach
giulia.malacarne@fmach.it

Kristof Engelen
CRI
Fondazione Edmund Mach
kristof.engelen@fmach.it

Andrea Argentini
Dpt. of Biochemistry, Ghent University
Dpt. of Medical Protein Research
VIB, Ghent

Valter Cavecchia
CNR-IMEM
Via alla Cascata 56/C Povo, Trento, Italy
valter.cavecchia@cnr.it

Claudio Moser
CRI, Fondazione Edmund Mach
S. Michele all'Adige, Italy
claudio.moser@fmach.it

Enrico Blanzieri
DISI, University of Trento
via Sommarive, 9 Povo, Trento, Italy
blanzier@disi.unitn.it

Abstract—**Our group has recently developed gene@home, a BOINC project that permits to search for candidate genes for the expansion of a gene regulatory network using gene expression data. The gene@home project adopts intensive variable-subsetting strategies enabled by the computational power provided by the volunteers who have joined the project by means of the BOINC client, and exploits the PC algorithm for discovering putative causal relationships within each subset of variables. This paper presents our TN-Grid infrastructure that is hosting the gene@home project. Gene@home implements a novel method for Network Expansion by Subsetting and Ranking Aggregation (NESRA), producing a list of genes that are candidates for the gene network expansion task. NESRA is an algorithm that has: 1) a ranking procedure that systematically subsets the variables; the subsetting is iterated several times and a ranked list of candidates is produced by counting the number of times a relationship is found; 2) several ranking steps are executed with different values of the dimension of the subsets and with different number of iterations producing several ranked lists; 3) the ranked lists are aggregated by using a state-of-the-art ranking aggregator. In our experimental results, we show that a single ranking step is enough to outperform both PC and PC*. Evaluations and experiments are done by means of the gene@home project on a real gene regulatory network of the model plant *Arabidopsis thaliana*.**

Keywords*—*Volunteer Computing; Distributed Computing; BOINC; Bioinformatics; Gene Network Expansion

I. Introduction

Gene expression data are accumulating at an increasing pace and compendia that integrate different data source are now available [26]. The causal relationships between the expression levels of genes, however, are still far from being completely characterized, even for model organisms. The information about the causal relationships between the gene expression levels can be organized in gene regulatory networks [14]. Methods that can guide and suggest possible candidates which regulate, or are regulated within a given gene network are important in the biological research where it is common to take into account prior knowledge about a phenomenon. In particular, considering the gene network that by knowledge or by hypothesis biologists assume to be relevant, an expansion method, for example GENESYS [33], can guide the discovery of candidate genes that can be causally connected to the network.

The PC algorithm [30], whose name derives from the initials of its authors is an algorithm that discovers causal relationships among variables. In particular, the PC algorithm is based on the systematic testing for conditional independence of variables given subsets of other variables. It has been comprehensively presented and evaluated by Kalish and colleagues [17] who proposed it also for gene network reconstruction purposes [22]. For this task, some modifications of the original formulation of the PC were also proposed by other authors [32], [31], [34], [36]. Existing other methods used for gene network reconstruction comprises ARACNE [25], [24], BANJO [13], and NIR [11]. Allen and colleagues [1] have recently compared ARACNE with other competitors in the task of large scale networks reconstruction and ARACNE proved to be a state-of-the-art method.

The task of gene network expansion is different and somehow more computationally demanding than performing a pure gene network reconstruction. A Local Gene Network (LGN) of an organism, is a subset of genes known to be causally connected. Gene network expansion can be informally defined as: given a LGN, find other candidate genes that are causally connected with the LGN. Once achieved a good reconstruction of the overall network it is in principle possible to use the information for deriving the expansion of a given subnetwork. However, the reconstruction should be done genome-wide with a considerable accuracy. On the contrary, in many gene network reconstruction tasks [23] the relevant genes to be included in the network are already known. When applied to genome-wide scale the reconstruction methods are computationally demanding and, as we will see here, not accurate enough for the expansion task.

In this paper, we explicitly define the task of finding candidates for gene network expansion and we propose a method that we called Network Expansion by Subsetting and Ranking Aggregation (NESRA) that is based on the PC algorithm and that we run on our project gene@home based on the BOINC platform [2]. We evaluate NESRA on real data of the model plant *Arabidopsis thaliana*.

The paper is organized as follows: Section II introduces the main ideas of our approach and in Section III we detailed both the TN-Grid platform and the gene@home project based on volunteer distributed computing. Section IV presents the NESRA algorithm, whose evaluation is described in Section V. Finally, Section VI draws some conclusions providing future insights for the gene@home project and the proposed methods.

II. Gene Network Expansion

Given a set S of gene transcripts whose level of expression has been measured p times such that for each $s_i \in S$ there is a vector $x_i \in \mathbb{R}^p$ of expression levels, and let us assume that there exists a golden truth direct graph $\mathcal{G} = (\mathcal{S}, \mathcal{B})$ with $\mathcal{B} \subset \mathcal{S} \times \mathcal{S}$ that represents the real causal relationships between the gene transcripts, it is possible to define the following tasks.

Task 1, Gene Network reconstruction. Given a subset of transcripts $N \subseteq S$, find a (direct) graph $G = (N, B)$ where $B \subset N \times N$ is a relation between the elements of N, and G approximates the subgraph in $\mathcal{G}$ obtained considering just the transcripts in N.

Task 2, Gene Network expansion. Given a graph $G = (N, B)$ where $N \subseteq S$ and $B \subset N \times N$ is a causal relation between the elements of N, find a graph $G' = (N', B')$ such that N' is a superset of N, B' is a superset of B, and G' approximates the subgraph $\mathcal{G}$ obtained considering just N'.

Task 3, Discovery candidate genes for Gene Network expansion. Given a graph $G = (N, B)$ where N is a subset of the transcripts of S and $B \subset N \times N$ is a relation between the elements of N, find a ranked list of elements of $S \setminus N$ such that the elements of the list are connected or very near to the elements of N in $\mathcal{G}$.

In our approach, the variables correspond to the genes and the samples are the actual measures, or measure comparisons against a reference, of the level of the expression in a given hybridization. In this paper, we will consider Task 3, motivated by the fact that in biological research the work is often guided by prior knowledge about the relevance of some genes. Moreover, a high-quality candidate short list would suffice because the actual validation of the possible interactions requires a complex mix of analytical and wet-lab techniques. It is worth to note that a perfect solution for Task 1 encompassing the whole genome would perfectly solve also Tasks 2 and Task 3 for all the possible networks. In the same way a perfect solution for Task 2 for a specific network would solve also Task 3. However, the state-of-the-art methods are far from perfect and a good solution for Task 3, in terms of precision of the candidate lists would be useful whenever the whole network and the interactions are still not known and Tasks 1 and 2 are not solved.

III. TN-Grid and the gene@home BOINC project

TN-Grid[1] is a BOINC server installation that has been thought and developed as an *umbrella* project, a service platform to give to local research groups a guided access to the power of the world-wide, volunteer-based, distributed BOINC [2] computing network. TN-Grid is the result of a joint effort made by two institutions of the Italian National Research Council (CNR), namely the Institute of Materials for Electronics and Magnetism (IMEM) and the Institute of Cognitive Sciences and Technologies (ISTC), both having local branches in Trento, Italy. At the time of writing, TN-Grid is the only public, BOINC-based active project in Italy.

The gene@home project is the first one hosted in the TN-Grid framework. It started as a collaboration between the Edmund Mach Foundation (FEM) and the Department of Information Engineering and Computer Science (DISI) of the University of Trento, Italy. The actual development of the gene@home project began as a course project in the academic year 2013/2014 during the Laboratory of Biological Data Mining course at the University of Trento, Italy. Gene@home is a distributed computational biology project based on the computation of the PC algorithm for the Gene Network Expansion task. The final goal of the project is the possibility to automatically perform Gene Network Expansion tasks.

After having setup our BOINC server, we coded several scripts to customize it accordingly to the needs of our project. In particular, we designed and developed the work generator using Python. BOINC APIs however, are available only through C++ libraries. For this reason, we implemented two C++ programs that wrap the necessary BOINC functions for the work generator. Our work generator is also responsible of the creation of the workunits that are then distributed to the volunteers. Because of this, the work generator has to predict the duration of each workunit. The duration of a workunit is not related to the execution time of a single PC run, its input data, or its parameters. So far, we are using a function that we obtained from a regression analysis on several workunits. However, as soon as we change the organism, the LGN, or the input data, we should re-do such analysis. To solve this issue we planned to have a benchmarking system able to estimate the duration of a workunit, making the work generator estimates more precise.

One of the most relevant parts of our implementation is the client application. The client application has been developed to be portable on a number of different architectures (32 and 64 bit) and operating systems, such as: Linux, Windows, and Mac OS. Our client application is actually a C++ implementation of the skeleton function (Algorithm 1), functionally equivalent to the one present in the "pcalg" R package [18], [15]. The choice of implementing the PC algorithm in C++ led to a speed-up of 240 times in the execution, together with a reduced memory

[1] http://gene.disi.unitn.it/test/

TABLE I: BOINC statistics of the gene@home project taken on four different days during the year 2014 and one time point in the year 2015 (reference period: the previous seven days). Date is represented as “dd/mm/yyyy”. *Over*: the total number of returned results, *Success*: successfully computed, *Valid*: validated results, *Initial*: pending validation, and *Errors*: faulty results.

Date	Over	Success (%)	Valid (%)	Initial	Errors
22/04/2014	15543	15392 (99.0%)	15340 (98.7%)	19	85
20/05/2014	69536	68621 (98.7%)	67096 (97.7%)	1450	89
16/12/2014	33232	31798 (96.2%)	29525 (88.8%)	2147	38
24/12/2014	91315	89536 (98.1%)	87584 (95.9%)	1716	61
27/03/2015	32062	30598 (95.4%)	29525 (92.1%)	865	34

consumption of about 10 times, when compared to the original version present in the R package. During the implementation and testing of the initial version of the gene@home project, we had to face several issues, mainly related to the characteristics of our project. One of them, in particular, is the amount of data that needs to be exchanged between the server and the users. We solved this problem with the help of the BOINC core developers that implemented the possibility of compressing the data during the upload and download phases. Subsequently, we optimized our implementation to further reduce the amount of data exchanged. When using a volunteer distributed system, one should be concerned about the validity of the results returned by the volunteers. On the gene@home server, we perform a validation step on the returned workunits, available in all BOINC systems. Because of the nature of our project, we were not able to find a self-validation method to confirm a result of a single workunit. For this reason, we are currently using a double validation method that consists of sending each workunit to two different volunteers. We then required the returned results to be equal bit-wise.

A first step of the processing of the results is implemented in the client application. As soon as a workunit finishes, a first aggregation of the results of the workunit is performed. This was also necessary in order to dramatically reduce the size of the output file that the volunteers need to upload in the gene@home server. The results collected with the gene@home project undergo further offline processing developed into a pipeline of Python and R scripts, that complete the analysis of the partial results of each workunit.

In Table I we present some statical results of the BOINC server collected in 5 different period of time. It is worth to note the high percentage values of successfully computed workunits, as well as the very low number of workunits that reported an error.

IV. NESRA

The general approach used by NESRA is to systematically and iteratively apply subsetting in order to compute several ranked lists with varying iterated subsetting parameters. The lists are then aggregated by means of a ranking aggregator. The high-level structure of NESRA is described in Algorithm 1. NESRA calls the ranking procedure (Algorithm 2) several times with different parameters producing several rankings that are then inputed to the ranking aggregation method for producing a final list.

The ranking procedure has three steps, which respectively create the subsets (Step 1), iterate several calls (Step 2) of the skeleton procedure of the PC algorithm (Algorithm 3) that processes the expression data of different subsets of the overall transcripts, and finally, compute the transcripts frequency that defines the order of each ranking (Step 3). The ranking procedure takes as parameters the number of iterations i and the dimension of the subset t as well as the significance level α for the PC algorithm. The computational cost of the PC algorithm is exponential, but it behaves reasonably in the case of sparse networks [22] consequently the ranking procedure requires relatively small values of t.

The ranking procedure is partially performed on the BOINC platform with the exception of the frequencies computation and the rankings aggregation that are implemented in Python and R scripts, which run outside BOINC.

A. Variable Subsetting

Subsetting is a computational practice that has been used in many domains including recently genomics [27]. It consists of selecting from the available data a subset of the data to be processed by the successive steps of the analysis. The idea in itself is not new and it can be found, with different names, at the very core of techniques such as bootstrapping or subsampling like in bagging [4] or singling-out features like in random forest [5] or in feature selection itself. We prefer here to call it subsetting for the sake of clarity because we will specifically focus on variable subsetting, namely different subsets of the variables will be used for gene network reconstruction using the PC algorithm. We avoid to call it subsampling because subsampling does not affect the presence of a variable but selects the samples of the variable. On the other hand, we do not call it feature selection because, in this setting the gene is not a feature that describes something, nor variable selection because we do not select variables in any way that is not purely random.

In NESRA subsetting is applied to genes to be selected for the application of the PC algorithm. The iterated subsetting will be systematic and controlled by two varying parameters producing a ranked list of genes for each pair of parameters values. Those rankings will be then aggregated.

B. Aggregation of ranked lists

The method that we propose as a solution, NESRA, exploits variable subsetting and ranking aggregation for tackle the problem formulated in Task 3.

We applied different ranking aggregation methods on the ranked lists. These methods are a simple technique such as the *number of appearances*, and less simple methods such as Borda Count [3] and MC4 heuristic [21], [9]. The baseline method that we considered is the *number of appearances* that counts in how many rankings a certain gene is present, i.e. the more a gene is present, the higher its position in the aggregated rank. The Borda Count method consists in constructing a matrix whose elements b_{ij} are for each gene s_i and ranking r_j the rank of the gene s_i in the ranking r_j. After that a statistic for every gene is computed on the rows of the matrix. The two statistical measures that we considered are the mean (BC-mean) and the minimum (BC-min) of the elements. MC4 heuristic is an aggregator based on Markov chains and it consists in computing a transition matrix such that the steady state of the chain assigns a higher probability to the elements with higher rank. MC4 has as parameter the significance level α_{MC4}.

C. The use of the gene@home project

NESRA exploits the gene@home project for computing the first two steps of the Algorithm 2. In details, the tiles creation (Step 1) is implemented in the work generator of the gene@home, while the application of the PC (Step 2) is implemented in the client application, running on the volunteer computers. A first aggregation of the results is then performed on the volunteer's computers, as soon as the workunit finishes. The complete processing of the results is then performed offline from BOINC by means of Python and R scripts.

Algorithm 1: NESRA.

```
Data: $S$ set of candidate transcripts, $S_{LGN}$ set of LGN
      transcripts, $E$ expression data
Input: $I$ set of values of number of iterations, $D$ set of
       values of the subset dimension, $A$ set of values
       of the significance level $\alpha$, $k$ maximum lenght
       of the lists
Result: ordered list of candidate transcripts
$L \leftarrow \emptyset$            // $L$ set of ordered lists
foreach $\alpha \in A$ do
    foreach $d \in D$ do
        foreach $i \in I$ do
            $L \leftarrow L \cup$ Ranking_Procedure($S$, $S_{LGN}$, $E$,
            $i$, $d$, $\alpha$)        // call Algorithm 2
$L \leftarrow$ top($L$, $k$)      // cut each list in $L$ to
the first $k$ elements
return Ranking_aggregation(L)
```

V. Evaluation of NESRA on *Arabidopsis thaliana*

In our evaluation of NESRA, we used the Flower Organ Specification Gene Regulatory Network (FOS) of the model plant *Arabidopsis thaliana*. The FOS gene network has been characterized and validated *in vivo* by the use of specific mutants [10], and it encompasses 15 genes (AT3G02310.1, AT1G69120, AT5G61850, AT1G30950, AT1G65480, AT5G15800, AT5G-60910, AT5G20240, AT4G36920, AT3G54340, AT2G17950, AT1G24260, AT5G11530, AT4G18960, AT5G03840.1) linked by 54 causal relationships [28]. Gene Expression Data for testing the algorithms were selected from the *A. thaliana* microarray expression data publicly available in the Plex database [8]. The dataset consists of 393 hybridization experiments of the GeneChip Arabidopsis ATH1 Genome Array that contains 22810 probe sets.

Algorithm 2: NESRA ranking procedure.

```
Data: $S$ set of candidate transcripts, $S_{LGN}$ set of LGN
      transcripts, $E$ expression data
Input: $i \geq 1$ number of iterations, $t$ subset dimension,
       $\alpha$ significance level
Result: $l$, ordered list of candidate transcripts
$N \leftarrow |S|$
$n \leftarrow |S_{LGN}|$
$L \leftarrow \emptyset$
foreach $g \in S$ do
    $p_g = i$
    $f_g = 0$
foreach $j$, $1 \leq j \leq i$ do
                // Step 1: tiles creation
    $S_{temp} \leftarrow S$
    foreach $h$, $1 \leq h \leq \lfloor N/t \rfloor$ do
        while $|T_{h,j}| < t$ do
            random select $g \in S_{temp}$
            $T_{h,j} \leftarrow T_{h,j} \cup \{g\}$
            $S_{temp} \leftarrow S_{temp} \setminus \{g\}$
    if $remainder(N/t) \neq 0$ then
        $h \leftarrow \lfloor N/t \rfloor$
        while $S_{temp} \neq \emptyset$ do
            random select $g \in S_{temp}$
            $T_{h+1,j} \leftarrow T_{h+1,j} \cup \{g\}$
            $S_{temp} \leftarrow S_{temp} \setminus \{g\}$
        while $|T_{h+1,j}| < t$ do
            random select $g \in S \setminus T_{h+1,j}$
            $T_{h+1,j} \leftarrow T_{h+1,j} \cup \{g\}$
            $pg \leftarrow pg + 1$
foreach $j$, $1 \leq j \leq i$ do
                // Step 2: PC application
    foreach $h$, $1 \leq h \leq \lceil N/t \rceil$ do
        $N_{h,j} = PC(T_{h,j}, E, \alpha)$    // call Algo 3
foreach $g \in S$ do
        // Step 3: Transcripts frequency
    computation
    foreach $q \in S_{LGN}$ do
        foreach $j$, $1 \leq j \leq i$ do
            foreach $h$, $1 \leq h \leq \lceil N/t \rceil$ do
                if $g \in Adj_{N_{h,j}}(q)$ then
                     // adjacent nodes of $q$ in
                    $N_{h,j}$
                    $l \leftarrow l \cup \{g\}$
                    $f_g \leftarrow f_g + 1$
    $f'_g = f_g/(p_g * n)$  // Normalized frequency
return 1 ordered w.r.t. $f'_g$
```

NESRA was run on the *A. thaliana* data as well as three competitors: PC, PC*, and ARACNE. The quality of the output list of NESRA and of the competitors was assessed by comparison with the available literature. A bibliographic search and classification of the genes provided in output by NESRA and by the competitors led to four classes: *Class 1*: genes reported to be biologically or functionally related to the LGN; *Class 2*: genes not reported to be directly related with the input network, but reported to be related with genes of Class 1; *Class 3*: genes described in literature, but reported not to be related with the input network or with the genes of Class 1; *Class 4*: genes not described in the available literature. A gene

Algorithm 3: PC Algorithm: skeleton procedure [17].

```
Data: T, Set of transcripts, $E$ expression data
Input: Significance level $\alpha$
Result: An undirected graph with causal relationship
        between transcripts
Graph $G \leftarrow$ complete undirected graph with nodes in T
$l \leftarrow -1$
while $l < |G|$ do
    $l \leftarrow l + 1$
    foreach $\exists u, v \in G$ s.t. $|Adj_G(u) \setminus \{v\}| \geq l$ do
        // $Adj_G(u)$ adjacent nodes of $u$ in $G$
        if $v \in Adj_G(u)$ then
            foreach $A \subseteq Adj_G(u) \setminus \{v\}$ s.t. $|A| = l$ do
                if $u, v$ are conditionally independent
                given $A$ w.r.t. $E$ with significance level $\alpha$
                then
                    remove edge $\{u, v\}$ from $G$
return G
```

TABLE II: *A. thaliana*, FOS network. Lists length and precision of the competitors, PC and PC*. PC values are mean and standard deviation of the 20 runs.

	Lists Length	Precision
PC, 20 runs	54.20 ± 1.28	0.40 ± 0.05
PC*	44	0.43

falling in Class 1 or Class 2 is considered to be a true positive and a gene in Class 3 or Class 4 a false positive. Precision is defined as the ratio between the number of true positives and the sum of true positives and false positives.

PC, PC*, and ARACNE solve the task of gene network reconstruction. For obtaining list of candidate genes for the expansion we considered all the genes that are connected to FOS genes in the resulting overall network. ARACNE was run with default parameters and the list was ranked according to the p-values that ARACNE itself provides. The PC algorithm was repeated 20 times shuffling the order of the input probe sets, given its dependency on the order. The results of the PC and PC* are reported in Table II, note that PC had a mean length of the list of 54.2 and so we took 55 as a cut-off for ARACNE and NESRA for sake of comparison. PC* found 44 genes, and since it is order independent we could not retrieve a result with 55 probes. We reported the result of ARACNE in Table IV because we used the p-values to evaluate the list at different cut-off values.

For NESRA we tried five different ranking aggregators: one based on the *number of appearances* in the 55-long lists used as baseline, two based on Borda Counts, BC-mean and BC-min, and then two based on the MC4 with two significance level values: $\alpha_{MC4} = 0.05$ and $\alpha_{MC4} = 0.01$.

The sets of parameters I, D, and A (see Algorithm 1) used by NESRA for numbers of iterations, subset dimensions, and the significance level are: $I = 100, 250, 500, 1000, 1500, 2000$, $D = 50, 100, 250, 500, 750, 1000, 1250, 1500, 1750, 2000$, and $A = 0.05$, respectively.

An example of the output list of a run of NESRA is shown in Table III, where we aggregated 60 different rankings. In order to assess the stability of NESRA we repeated it 30 times with selected parameters, mean and standard deviations of the results are presented in Table IV.

MC4 and BC-mean present in general very good results. BC-min instead, gives more variable outputs, sometimes showing better results ($k = 5$), but in other cases behaving as the baseline method ($k = 20$ or $k = 55$). The results in Table IV show also that, regardless of the aggregation method used, NESRA founds more correct genes (genes belonging to either Class 1 or Class 2) in the first 20 positions ($k = 5, 10, 20$) compared to ARACNE. ARACNE instead, founds correct genes only when considering a longer list ($k = 55$).

VI. Conclusions

We have presented the TN-Grid platform that is hosting the gene@home BOINC project. In particular, the gene@home project has been developed with the idea of automatically perform the Gene Network Expansion task. The gene@home project, so far, is running only on the CPUs of the volunteers' computers. As a future improvement of the gene@home, we developed and tested a parallel version of PC* for executing on the Graphics Processing Unit (GPU). The choice of implementing PC* instead of PC is due to its independence with respect to the order of the input.

We also presented NESRA that is a new method that exploits variable subsetting and ranking aggregation to find candidate genes for the expansion of gene networks. The method relies on the BOINC platform for running the PC algorithm while all the other post-processing, ranking and aggregation analyses, are performed offline. The evaluation on the FOS gene network of the model plant *Arabidopsis thaliana* shows good results, and when the results are compared to the biological literature, NESRA outperforms the competitors. In general, NESRA can be used to find candidate variables that are causally connected to other variables and it has proved to work with more than 20000 variables. We foresee the application of NESRA also in other biological domains.

References

[1] J. D. Allen, Y. Xie, M. Chen, L. Girard, and G. Xiao. Comparing statistical methods for constructing large scale gene networks. *PLoS ONE*, 7(1):e29348, 2012.

[2] D. P. Anderson. BOINC: A system for public-resource computing and storage. In *Proceedings of the 5th IEEE/ACM International Workshop on Grid Computing*, GRID '04, pages 4–10, Washington, DC, USA, 2004. IEEE Computer Society.

[3] J. Borda. *Mémoire sur les élections au Scrutin*. Histoire de l' Académie Royale des Sciences, 1781.

[4] L. Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996.

[5] L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

[6] X. Cai et al. A putative CCAAT-binding transcription factor is a regulator of flowering timing in Arabidopsis. *Plant Physiol.*, 145(1):98–105, Sep 2007.

[7] L. Q. Chen et al. Sucrose efflux mediated by SWEET proteins as a key step for phloem transport. *Science*, 335(6065):207–211, Jan 2012.

[8] S. Dash, J. Van Hemert, L. Hong, R. P. Wise, and J. A. Dickerson. PLEXdb: gene expression resources for plants and plant pathogens. *Nucleic Acids Res.*, 40(Database issue):D1194–1201, Jan 2012.

TABLE III: *A. thaliana*, FOS network. Example of output list of NESRA with ranking aggregation method MC4 with $\alpha_{MC4} = 0.01$ with precision 0.90 with $k = 5$ and 0.80 with $k = 10$.

Rank	AffyID	Gene	Annotation	Class
1	259089_at	AT3G04960	similar to unknown protein	Class 1 [20]
2	255644_at	AT4G00870	basic helix-loop-helix (bHLH) family protein	Class 2 [16]
3	265441_at	AT2G20870	cell wall protein precursor	Class 1 [6]
4	267528_at	AT2G45650	AGL6 (AGAMOUS LIKE-6)	Class 1 [35]
5.5	245571_at	AT4G14695	unknown protein	Class 4
5.5	249939_at	AT5G22430	similar to unknown protein	Class 1 [37]
7	245842_at	AT1G58430	RXF26	Class 1 [29]
8	248496_at	AT5G50790	ATSWEET10	Class 3 [7]
9	264180_at	AT1G02190	CER1 protein	Class 1 [12]
10	261375_at	AT1G53160	SPL4 (SQUAMOSA PROMOTER BINDING PROTEIN-LIKE 4)	Class 1 [19]

TABLE IV: *A. thaliana*, FOS network. NESRA precisions (mean and standard deviation) on 30 different runs with the sets of parameters: values of iterations $I' = 100, 500, 2000$ and subset dimensions $D' = 1000, 2000$.

Aggregation Method	k=5	k=10	k=20	k=55
N of appearances	0.54 ± 0.054	0.54 ± 0.054	0.53 ± 0.060	0.42 ± 0.015
BC-mean	0.90 ± 0.098	0.65 ± 0.049	0.63 ± 0.038	0.43 ± 0.016
BC-min	0.86 ± 0.098	0.68 ± 0.038	0.60 ± 0.053	0.43 ± 0.021
MC4 ($\alpha_{MC4} = 0.05$)	0.88 ± 0.098	0.65 ± 0.049	0.63 ± 0.038	0.42 ± 0.012
MC4 ($\alpha_{MC4} = 0.01$)	0.88 ± 0.098	0.65 ± 0.049	0.63 ± 0.038	0.42 ± 0.012
ARACNE	0.20	0.30	0.35	0.45

[9] C. Dwork, R. Kumar, M. Naor, and D. Sivakumar. Rank Aggregation Methods for the Web. In *Proceedings of the 10-th WWW Conference*, pages 613–622, 2001.

[10] C. Espinosa-Soto et al. A gene regulatory network model for cell-fate determination during Arabidopsis thaliana flower development that is robust and recovers experimental gene expression profiles. *Plant Cell*, 16(11):2923–2939, Nov 2004.

[11] T. S. Gardner et al. Inferring genetic networks and identifying compound mode of action via expression profiling. *Science*, 301(5629):102–105, Jul 2003.

[12] C. Gomez-Mena, S. de Folter, M. M. Costa, G. C. Angenent, and R. Sablowski. Transcriptional program controlled by the floral homeotic gene AGAMOUS during early organogenesis. *Development*, 132(3):429–438, Feb 2005.

[13] A. J. Hartemink. Reverse engineering gene regulatory networks. *Nat. Biotechnol.*, 23(5):554–555, May 2005.

[14] J. Hasty et al. Computational studies of gene regulatory networks: in numero molecular biology. *Nat. Rev. Genet.*, 2(4):268–279, Apr 2001.

[15] A. Hauser and P. Bühlmann. Characterization and greedy learning of interventional Markov equivalence classes of directed acyclic graphs. *Journal of Machine Learning Research*, 13:2409–2464, 2012.

[16] W. Hu, Y. Wang, C. Bowers, and H. Ma. Isolation, sequence analysis, and expression studies of florally expressed cDNAs in Arabidopsis. *Plant Mol. Biol.*, 53(4):545–563, Nov 2003.

[17] M. Kalisch and P. Bühlmann. Estimating high-dimensional directed acyclic graphs with the PC-algorithm. *J. Mach. Learn. Res.*, 8:613–636, May 2007.

[18] M. Kalisch, M. Mächler, D. Colombo, M. H. Maathuis, and P. Bühlmann. Causal inference using graphical models with the R package pcalg. *Journal of Statistical Software*, 47(11):1–26, 2012.

[19] S. Lal, L. B. Pacis, and H. M. Smith. Regulation of the SQUAMOSA PROMOTER-BINDING PROTEIN-LIKE genes/microRNA156 module by the homeodomain proteins PENNYWISE and POUND-FOOLISH in Arabidopsis. *Mol Plant*, 4(6):1123–1132, Nov 2011.

[20] J. Y. Lee, S. F. Baum, J. Alvarez, A. Patel, D. H. Chitwood, and J. L. Bowman. Activation of CRABS CLAW in the Nectaries and Carpels of Arabidopsis. *Plant Cell*, 17(1):25–36, Jan 2005.

[21] S. Lin. Rank Aggregation Methods. *Wiley Interdisciplinary Reviews: Computational Statistics*, 2(5):555–570, 2010.

[22] M. H. Maathuis et al. Predicting causal effects in large-scale systems from observational data. *Nat. Methods*, 7(4):247–248, Apr 2010.

[23] D. Marbach et al. Wisdom of crowds for robust gene network inference. *Nat. Methods*, 9(8):796–804, Aug 2012.

[24] A. A. Margolin et al. ARACNE: an algorithm for the reconstruction of gene regulatory networks in a mammalian cellular context. *BMC Bioinformatics*, 7 Suppl 1:S7, 2006.

[25] A. A. Margolin et al. Reverse engineering cellular networks. *Nat Protoc*, 1(2):662–671, 2006.

[26] P. Meysman et al. COLOMBOS v2.0: an ever expanding collection of bacterial expression compendia. *Nucleic Acids Res.*, 42(Database issue):D649–653, Jan 2014.

[27] L. A. Peternelli. Cost-effective implementation of genomic selection by subsetting snp markers. In *Plant and Animal Genome XXIII Conference*. Plant and Animal Genome, 2015.

[28] Y. E. Sanchez-Corrales et al. The Arabidopsis thaliana flower organ specification gene regulatory network determines a robust differentiation process. *J. Theor. Biol.*, 264(3):971–983, Jun 2010.

[29] J. X. Shi et al. SHINE transcription factors act redundantly to pattern the archetypal surface of Arabidopsis flower organs. *PLoS Genet.*, 7(5):e1001388, May 2011.

[30] P. Spirtes and C. Glymour. An algorithm for fast recovery of sparse causal graphs. *Social Science Computer Review*, 9:62–72, 1991.

[31] M. Tan, M. Alshalalfa, R. Alhajj, and F. Polat. Influence of prior knowledge in constraint-based learning of gene regulatory networks. *IEEE/ACM Trans Comput Biol Bioinform*, 8(1):130–142, 2011.

[32] M. Tan et al. Combining multiple types of biological data in constraint-based learning of gene regulatory networks. In *Computational Intelligence in Bioinformatics and Computational Biology, 2008. CIBCB '08. IEEE Symposium on*, pages 90–97, Sept 2008.

[33] A. Tanay and R. Shamir. Computational expansion of genetic networks. *Bioinformatics*, 17 Suppl 1:S270–278, 2001.

[34] M. Wang et al. Inferring large-scale gene regulatory networks using a low-order constraint-based algorithm. *Mol Biosyst*, 6(6):988–998, Jun 2010.

[35] S. L. Yoo et al. AGAMOUS-LIKE 6 is a floral promoter that negatively regulates the FLC/MAF clade genes and positively regulates FT in Arabidopsis. *Plant J.*, 65(1):62–76, Jan 2011.

[36] X. Zhang et al. Inferring gene regulatory networks from gene expression data by path consistency algorithm based on conditional mutual information. *Bioinformatics*, 28(1):98–104, Jan 2012.

[37] M. Zik and V. F. Irish. Global identification of target genes regulated by APETALA3 and PISTILLATA floral homeotic gene action. *Plant Cell*, 15(1):207–222, Jan 2003.