Cis-regulatory Module Detection using Constraint Programming

Tias Guns[1], Hong Sun[2,3], Kathleen Marchal[3], Siegfried Nijssen[1]
[1] Department of Computer Science, Katholieke Universiteit Leuven, Belgium.
email: {tias.guns,siegfried.nijssen}@cs.kuleuven.be
[2] Department of Microbial and Molecular systems, Katholieke Universiteit Leuven, Belgium.
email: hong.sun@esat.kuleuven.be, kathleen.marchal@biw.kuleuven.be
[3] Department of Electrical Engineering, Katholieke Universiteit Leuven, Belgium.
email: hong.sun@esat.kuleuven.be

***Abstract*—We propose a method for finding CRMs in a set of *co*-regulated genes. Each CRM consists of a set of binding sites of transcription factors. We wish to find CRMs involving the same transcription factors in multiple sequences. Finding such a combination of transcription factors is inherently a combinatorial problem. We solve this problem by combining the principles of itemset mining and constraint programming. The constraints involve the putative binding sites of transcription factors, the number of sequences in which they co-occur and the proximity of the binding sites. Genomic background sequences are used to assess the significance of the modules. We experimentally validate our approach and compare it with state-of-the-art techniques.**

***Index Terms*—*cis*-regulatory module, itemset mining, constraint programming.**

I. Introduction

The detection of transcription factor binding sites is key in developing a better understanding in the regulation of the genome, and hence many tools have been developed to address this problem. An important class of such tools deals with the problem of finding transcription factor binding sites of motifs in a single given sequence, see e.g. [1] for an overview and comparison. Although successful to a certain degree, the performance of these methods suffers by the prediction of many false positive binding sites [1]. Key in avoiding these false positives is the exploitation of additional knowledge, if available. In this paper we study one such case, in which we assume several co-regulated genes given. We are interested in finding a set of motifs having *cis*-regulatory modules (CRMs) in multiple of these sequences. There are several features that could allow us to reduce the number of false positive predictions in this case:

- we are only interested in sets of motifs that have binding sites in each other's proximity; this excludes binding sites scattered over the genomic sequence;
- we are only interested in sets of motifs that are common to multiple sequences; this excludes sets of motifs that are specific to one sequence;
- large numbers of genomic sequences are available that allow the statistical assessment of the identified CRMs.

Ideally, we would like to exploit all these features in one algorithm. However, few algorithms are capable of this. Several techniques have been proposed for the discovery of *cis*-regulatory modules [2], [3], [4], but they do not consider multiple sequences. Other tools do not take into account the proximity constraint [5], [6], or differ in the evaluation of the CRMs [7].

In this work, we propose a general approach which takes all these 3 features into account. The approach is based on combining itemset mining with the general methodology of constraint programming. Itemset mining, a technique developed in the data mining community, has been used for similar problems before [5], [6], [7], but our technique differs in several ways from these earlier techniques. First, most itemset mining techniques do not take into account proximity of motif hits [5], [6]. Second, we use a novel approach to eliminate redundant sets of motifs; this makes our algorithm more efficient and renders its output more useful. It also allows us to deal with both large numbers of motifs and large numbers of sequences. Finally, in order to be able to deal with this wide range of requirements, we build on the general principles of constraint programming, which allows us to combine these requirements more flexibly.

In this paper, we will first introduce the general idea of our approach, followed by a discussion of how our earlier work on combining itemset mining and constraint programming [8] can be extended to CRM detection. Our system, called CPModule, is finally validated by experiments.

II. Method Overview

Our method consists of 3 phases: screening, mining and ranking. In the remainder of this section we provide an overview of the tasks performed in these phases. Our main contribution is in the calculations performed in the mining phase. Details of this phase are provided in the next section.

A. *Phase 1: Screening*

Like most other module detection methods [3], [4], [7], [9], our method starts from an existing library of motif models for transcription factors, in this case position weight matrices (PWMs). The genomic sequences under investigation are screened with these PWMs to identify putative transcription factor binding sites (TBFSs). The result of this phase is hence

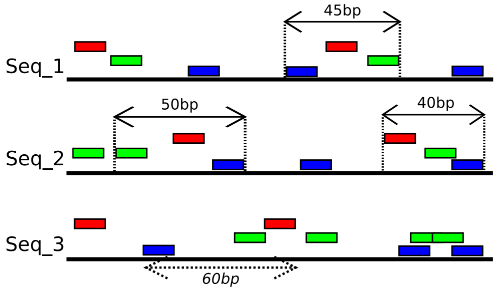

Fig. 1. Proximity of a motif set in 3 genomic sequences (distance = 50)

for each motif M and sequence S a set of motif hit intervals

$$MH(M,S) = \{(l,r) \mid 1 \leq l < r \leq |S|, M \text{ has a hit at } (l,r)\};$$

here (l,r) is an interval between positions l and r on the sequence. To identify hits, many alternative systems can be used; we will use Clover [10], which assigns a raw score to each hit; by applying a threshold, we can identify none, one or multiple hits per motif and per genomic sequence, depending on the threshold applied.

B. Phase 2: Mining

This phase takes the motif hits in the target sequences, found in the screening phase, as input. We use these hits to search for potential CRMs across multiple sequences. A potential CRM is characterized by a set of motif hits appearing in each other's proximity on a sequence. If there is a region in the sequence in which each of the motifs has at least one hit, we say that they are in each others proximity. The maximal distance θ of that region is specified by the user and controls the level of proximity. More formally, a set of motifs $\mathcal{M} = \{M_1, \dots M_n\}$ is a potential CRM in a sequence S iff its set of hit regions is not empty, where the set of hit regions is defined as follows:

$$\begin{aligned} HR(\mathcal{M},S) = \{(l, l+\theta) \mid 1 \leq l \leq |S|, \forall M \in \mathcal{M} : \\ \exists (l',r') \in MH(M,S) : l \leq l' < r' \leq l+\theta\}. \end{aligned}$$

An example is shown in Figure 1. Given a maximum distance of 50bp, the Red, Green and Blue motif sets are within each other's proximity in sequence 1 and 2. In sequence 2, there are two regions containing hits of the 3 motifs. In sequence 3, the smallest region containing all 3 motifs has a distance of 60bp. Hence, the motifs are not within each other's proximity in that sequence. Given a set of sequences $\mathcal{S}$, the subset of sequences in which a set of motifs $\mathcal{M}$ forms a potential CRM is denoted by $\varphi(\mathcal{M}, \mathcal{S})$.

A key element of our approach is that we are looking in a large collection of motifs for representative motif sets across multiple sequences. We formalize this by requiring that a minimal number of the input sequences contains the motif set as a potential CRM, i.e. $|\varphi(\mathcal{M},\mathcal{S})| \geq \mathit{min\_frequency}$, for some threshold $\mathit{min\_frequency}$. Hence our mining algorithm enumerates all combinations of motifs that have hits in each other's proximity, in a sufficient number of genomic sequences. For this task we use a constraint programming technique that will be discussed in detail later.

The output of this phase is hence a list of motif sets $\mathcal{M}$, each a potential CRM. An important issue is that of redundancy in the collection of motif sets. It is possible that two motif sets $\mathcal{M}_1$ and $\mathcal{M}_2$, where $\mathcal{M}_1 \subset \mathcal{M}_2$, are found in the same set of sequences, i.e. $\varphi(\mathcal{M}_1, \mathcal{S}) = \varphi(\mathcal{M}_2, \mathcal{S})$. To improve the efficiency of the search and reduce the size of the input of the next phase, it can be desirable not to list all redundant sets. Our framework allows us to deal with this issue.

C. Phase 3: Ranking

To assess the significance of each of the potential CRMs found in the previous phase, we determine their statistical significance. We want to find motif sets which are very specific to our target genomic sequences, but not to a background model [11]. The background model consists of a large number of random samples of sequences from the genome. We use it to calculate a p-value, and rank the potential CRMs accordingly.

To calculate the p-value, we adapt the strategy proposed in [12]. We compare the number of observed sequences that contain the motif set, $|\varphi(\mathcal{M}, \mathcal{S})|$, with the expected number of sequences. The latter is estimated by counting the number of background sequences containing the motif set, where we use exactly the same screening as on the target sequences to calculate $|\varphi(\mathcal{M}, \mathcal{S}_{background})|$. From this, we calculate a p-value by means of a binomial distribution:

$$p - value(\mathcal{M}) = \sum_{i=|\varphi(\mathcal{M},\mathcal{S})|}^{|\mathcal{S}|} \binom{|\mathcal{S}|}{i} p^i (1-p)^{|\mathcal{S}|-i};$$

where $p = |\varphi(\mathcal{M}, \mathcal{S}_{background})| / |\mathcal{S}_{background}|$; $\mathcal{S}$ is the set of target sequences; $\mathcal{S}_{background}$ the set of background sequences.

Note that in this ranking, for two motif sets $\mathcal{M}_1 \subseteq \mathcal{M}_2$, with $\varphi(\mathcal{M}_1, \mathcal{S}) = \varphi(\mathcal{M}_2, \mathcal{S})$, motif set $\mathcal{M}_2$ will never score worse than $\mathcal{M}_1$ and is hence of more interest. Avoiding redundancies in phase 2 hence makes the ranking more useful.

III. Mining Motif Sets

As pointed out in the previous section, we need to find a set of motifs fulfilling a wide range of conditions. To solve this combinatorial problem, in previous work *itemset mining* algorithms were used [5], [6], [7]. Many itemset mining algorithms are known in the literature, each of which was usually developed to solve one specific task. To address the task in this paper, we could also develop a specialized itemset mining algorithm. The main drawback of this approach is that it does not provide indications on how to extend the algorithm in the future if further extensions are needed. Hence we propose to use a more general approach, in which constraints can be seen as components of a large, general system and there is a clear way of adding constraints in the system. This modular approach also allows us to combine the constraints we are currently faced with in a principled and correct way.

The approach that we propose is based on the framework of constraint programming for itemset mining (CP4IM) [8]. Constraint programming is an area of research which studies how to solve general combinatorial problems. It has been applied on many challenging tasks such as scheduling, planning and more recently, itemset mining. Its approach can be summarized by the tag line

constraint programming = model + search.

The *model* is a specification of a problem in terms of constraints, given by the user. It can be formalized as a triplet (V, D, C) consisting of variables V, a domain $D(v)$ of possible values for each variable $v \in V$, and a set of constraints C. Each constraint is defined on a set of variables. A solution to the model is an assignment of one value to each variable that satisfies all the constraints. The search is done by a generic solver that uses the constraints to enumerate only valid solutions. Hence, from a user's perspective, an advantage is that the solution strategy is hidden and the user only has to study how to formalize a problem in a model.

The *search* strategy taken by a CP solver is based on depth-first backtracking search. It is an alternation of branching, in which a variable is assigned a value from its domain, and propagation. Propagation is the process of using a constraint to remove values from the domain of variables that would violate it. Every constraint has a corresponding propagator that does its propagation. As a simple example: for constraint $x+y \geq 2$ and initial domains $D(x) = D(y) = \{0, 1\}$, the corresponding propagator would remove value 0 from x and y's domains.

The constraint programming approach is modular as it is straightforward to add constraints; essentially, constraints are added by adding propagators. All constraints present in the system can freely be combined as there is no other interaction between constraints than through the variables influenced by the propagators. We exploit this modularity in this paper by adding a new constraint in an existing CP system [13], which is sufficient to deal with the proximity requirement.

In the following, we propose a constraint programming model for the CRM detection problem, consisting of 4 different constraints which we describe in turn.

First, we identify the variables in our model. We propose a CSP formulation in which there is a boolean variable $\widetilde{M}_i$ for every motif, indicating whether this motif is part of the motif set. If a certain $\widetilde{M}_i = 1$, then we say that the motif is in the motif set; otherwise the motif is not in the set. Furthermore, we have a boolean variable $\widetilde{S}_j$ for every genomic sequence, indicating whether the motif set is a potential CRM in a sequence, i.e. whether $S_j \in \varphi(\mathcal{M})$. Lastly, we define a boolean variable $\widetilde{seqM}_{ij}$ for every motif i and every sequence j. The variables $\widetilde{seqM}_{ij}$ indicate whether motif M_i is in the proximity of the motifs in motif set $\mathcal{M}$ on sequence j (we will define this more formally below).

The following constraints are imposed on these variables:

1) Frequency Constraint: The constraint that imposes a minimum size on $\varphi(\mathcal{M})$ is easily formalized as:

$$\sum_j \widetilde{S}_j \geq \mathit{min\_frequency}.$$

Constraint programming system provide a propagator for this constraint by default.

2) Proximity Constraint: The essential constraint is the proximity constraint, which will couple the $\widetilde{seqM}_{ij}$ variables to the variables representing motifs. Formally, we define the $\widetilde{seqM}_{ij}$ variables as follows for every motif on every genomic sequence:

$$\begin{aligned}\forall ij : \widetilde{seqM}_{ij} = 1 \Leftrightarrow (\exists (l, r) \in HR(\mathcal{M}, S_j) \\ \exists (l', r') \in MH(M_i, S_j) : l \leq l' < r' \leq r). \quad (1)\end{aligned}$$

In other words, if in a particular genomic sequence a particular motif is within a hit region of the motif set, this motif's variable for that sequence must be 1. Readers familiar with itemset mining may notice that $seqM_{ij}$ can be thought of us a binary matrix as in traditional itemset mining; however, here this matrix is dynamically defined by means of additional constraints. Observe that $\widetilde{seqM}_{ij} = 1$ will hold for all motifs in the motif set $\mathcal{M}$, for all sequences that are in $\varphi(\mathcal{M})$; however, there may be additional motifs that have hits in the proximity of regions in $HR(\mathcal{M}, S_j)$. Constraint (1) is a specialized constraint for our problem for which no efficient propagator is available in existing CP systems. Hence, we need to propose a new propagator, which we present later in this section.

3) Coverage Constraint: Using the variables defined by the proximity constraint, we can define the $\widetilde{S}$ variables as follows:

$$\forall j : \widetilde{S}_j = 1 \Leftrightarrow \forall i : \widetilde{M}_i = 0 \vee \widetilde{seqM}_{ij} = 1.$$

This binary constraint is already available in most CP systems.

4) Redundancy Constraint: As we pointed out earlier, exhaustive search is likely to consider a large number of solutions, some of which can be considered redundant with respect to each other. This is partly due to the non sparsity of the data (data typically consists of multiple binding sites for most motif and sequence combination). For instance, if a motif set consisting of 5 motifs $\{a, b, c, d, e\}$ meets the proximity and frequency constraints, then any of its subsets $\{a, b, c, d\}, \{a, b, c, e\}, \ldots, \{b, c, e\}, \ldots, \{e\}$ will also contribute to CRMs that meet the same constraints and hence will be reported as a solution. Many of those subsets contribute to CRMs that occur in exactly the same sequences and often contain exactly the same binding regions as those identified for the larger superset. Tests on small datasets indicated that up to 80% of the solutions, and hence computation time, is spent on enumerating redundant solutions. To avoid these solutions, we imposed that a solution has to be maximally specific given the sequences that it covers. More formally, we require that

$$\forall i : \widetilde{M}_i = 1 \Leftrightarrow \forall j : \widetilde{S}_j = 0 \vee \widetilde{seqM}_{ij} = 1.$$

Note that this constraint is almost identical to the constraint for traditional closed itemset mining [8]. The main difference is that the constraint is defined over the binary $\widetilde{seqM}_{ij}$ *variables*, whereas in traditional closed itemset mining the binary matrix is constant. Consequently, certain types of propagation which are always possible in traditional closed itemset mining, are only possible when the domains of the variables $\widetilde{seqM}_{ij}$ are fixed. For instance, whereas in traditional closed itemset mining, if two motifs always occur in the same sequences, we can always add them together, in our case, the proximity constraint may disallow this; the hits of the motifs may be too far from each other for the motifs to be added together in a set. This makes the adaption of traditional itemset mining algorithms difficult. However, the implementation of a correct propagator for the proximity constraint, as well as the modular combination of this propagator with propagators for other constraints ensures that the search is still performed correctly and efficiently in CP, without having to implement a new itemset mining algorithm from scratch.

Propagation of the Constraints

The frequency, coverage and redundancy constraints are variants of the ones used in constraint programming for itemset mining. We refer to [8] for details. Here we focus on the propagation of the proximity constraint (1), which is an essential component of our proposed approach. Essentially, we need to propagate changes of the motif variables to the $\widetilde{seqM}_{ij}$ variables. Initially, the domains of the $\widetilde{seqM}_{ij}$ and $\widetilde{M}_i$ variables are $\{0, 1\}$. The propagator does the following:

a) Remove 1: The value 1 in the domain of $\widetilde{seqM}_{ij}$ indicates that at this point of the search it is still possible for motif i to appear in the proximity of the final CRM. Value 1 is removed when it does not hold that:

$$\exists (l, r) \in HR(\mathcal{M}, S_j) \quad \exists (l', r') \in MH(M_i, S_j) : l \leq l' < r' \leq r,$$

where $\mathcal{M} = \{M_j \,|\, D(\widetilde{M}_j) = \{1\}\}$ represents the motifs that have been fixed in the search. In words, when a motif does not have a hit in the proximity of motifs already fixed, we remove it from consideration, as only motifs close enough to other motifs can be part of a potential CRM.

b) Remove 0: The value 0 in the domain of $\widetilde{seqM}_{ij}$ represents the possibility that there is a solution in which no hit of motif i in sequence j is part of a CRM. However, if the largest motif set still reachable actually has a hit region in sequence j fulfilling the proximity constraint, and motif i is within this region, we may conclude that motif i will always be part of the CRM. In other words, we remove 0 if:

$$\exists (l, r) \in HR(\mathcal{M}, S_j) \quad \exists (l', r') \in MH(M_i, S_j) : l \leq l' < r' \leq r),$$

where $\mathcal{M} = \{M_j \,|\, 1 \in D(\widetilde{M}_j)\}$.

The removal of zeros is in particular important as variables fixed to 1 ($D(\widetilde{seqM}_{ij}) = \{1\}$) can be exploited to propagate other constraints; when a sufficient number of motifs is fixed, it may be concluded that certain other motifs will always be part of the CRM.

IV. Experiments

In this section we evaluate the effectiveness of our approach by answering the following three research questions:

- Q1. How well does our algorithm improve the results of the screening phase?
- Q2. What is the effect of the proximity constraint on the quality of the result?
- Q3. How does our method compare to state-of-the-art methods?

To answer these questions we use data constructed by Xie et al. 2008 [14]. The data consists of 22 genomic sequences, each 1000 base pairs in length. In the first 20 sequences, transcription factors Oct4, Sox2 and FoxD3 are each inserted 3 times, in a region of at most 164bp. The inserted nucleotides are sampled from the respective TRANSFAC PWMs. The last two sequences have no inserted transcription factors[1].

All the methods under evaluation start from a given set of motif models. We use 516 transcription factors (TFs) from the TRANSFAC database [15], obtained after filtering all 584 vertebrate TFs using the motif similarity checking tool MotifComparison [16] with a threshold of 0.1.

We measure the quality of a method's prediction by comparing the best solution found with the inserted modules. In all methods, a solution consists of at most one predicted set of motifs for every sequence, with for every motif a number of hits at which it is predicted to bind. Recurring motif detection tools predict one set of motifs with hits in multiple sequences. Often a binding region for the whole motif set is returned instead of hits for each motif separately.

As in [17], we evaluate both at the motif level and at the nucleotide level. We compare different solutions by comparing their contingency tables. At the motif level, a predicted motif for a sequence is a true positive (TP) if that motif was indeed inserted in that sequence, otherwise it is a false positive (FP). If a motif was not predicted, but was inserted in that sequence, it is counted as a false negative (FN), otherwise as a true negative (TN). As the motif-level evaluation does not take the predicted binding sites into account, we also evaluate a solution at the nucleotide level: for every nucleotide we verify whether it was predicted to be part of the CRM and whether it should have been predicted or not, again resulting in TP, FP, FN, and TN counts. These counts are aggregated over all sequences to obtain the total counts of this solution. Ideally, a solution scores good at both the motif and nucleotide level.

In the following experiments we used the Clover tool [10] in the screening phase. The default thresholds were used unless mentioned otherwise and no randomization options were used. In the mining phase we use a proximity threshold of 165 bp and require a minimum frequency of 60%. There is no limit on the maximum number of motifs that can be selected. In the ranking phase we use 2000 randomly selected non-repeating intergenic sequences from the Mouse genome.

CPModule predicts that a nucleotide at position k in a sequence S is positive when for the highest ranked motif set $\mathcal{M}$ there exists a $(l, r) \in HR(\mathcal{M}, S)$ with $l_\top \leq k \leq r_\perp$. Here $l_\top = \min\{l' \,|\, M_i \in \mathcal{M}, (l', r') \in MH(M_i, S), l' \geq l\}$ and $r_\perp = \max\{r' \,|\, M_i \in \mathcal{M}, (l', r') \in MH(M_i, S), r' \leq r\}$.

A. Improvements over the Screening Phase

To answer Q1, we compare the quality of the motifs and hits predicted by the screening tool in phase 1 with the best

[1] Datasets and CPmodule software (based on the Gecode Constraint Programming system) available at http://dtai.cs.kuleuven.be/CP4IM/CPmodule/

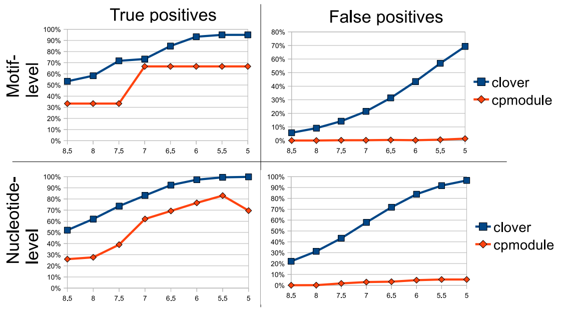

Fig. 2. Comparison of the single motif scanning tool with the output of our method (phase 3); on the x axis is the raw threshold provided to the single motif scanning tool

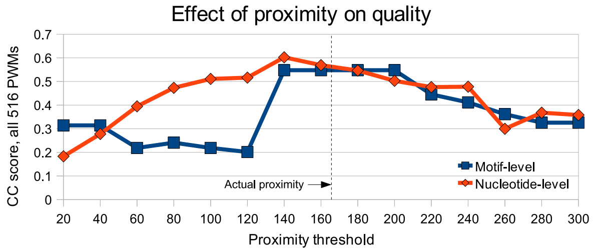

Fig. 3. Effect of the proximity constraint on quality

module predicted after phase 3. In the screening phase we vary the threshold on the hit score. A lower score results in more hits of lower quality. Figure 2 lists the relative number of true and false positives at both the motif level (top row) and nucleotide level (bottom row) for different hit score thresholds. At the motif level, when decreasing the hit score, the screening tool (blue line) predicts more true motifs correctly, however the number of falsely predicted motifs increases at a far larger rate. This confirms earlier findings [1] that single motif detection tools have a large number of false positives. In contrast, our method (red line) finds many true positives while including only a fraction of the false positives. Even when the screening tool returns more than 50% false positives at the motif level, our algorithm selects at most 1% false positives. The situation is similar at the nucleotide level: our algorithm returns a solution that has many true positives, yet only a fraction of the false positives. Note that with a hit score of 5 the performance of our algorithm decreases. This relates to the fact that at this threshold, the density of hits gets so large that an increasing number of motifs will have a hit in each other's proximity matching the algorithm's criterion. This leads to a considerable decrease in predictive and computational performance.

B. Effect of the Proximity Constraint

To answer Q2, we run our method using different proximity threshold values. Figure 3 shows the motif level and nucleotide level correlation coefficient, calculated as in [17]. The true proximity value of 164bp is indicated by a vertical line. The binding region predicted by our algorithm is by definition at most the size of the proximity threshold, hence at a low threshold the nucleotide level score is low and modules with at most one true motif are found. Using thresholds between 140 and 200bp our method achieves its highest motif level score, but the method is not sensitive to the exact setting of the parameter within this range. The nucleotide level score is more sensitive to the proximity threshold as a larger threshold results in larger predicted binding regions and hence more false positively predicted nucleotides.

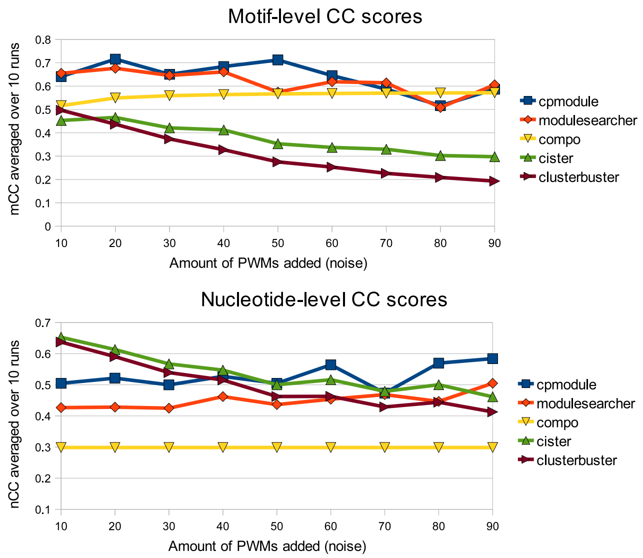

Fig. 4. Comparing motif and nucleotide level correlation coefficients of several algorithms

TABLE I
MOTIF AND NUCLEOTIDE LEVEL CC SCORES FOR DIFFERENT ALGORITHMS USING ALL 516 TRANSFAC PWMS

	cister	clusterbuster	modulesearcher	compo	cpmodule
mCC	0.16	0.05	/	-	**0.57**
nCC	0.23	0.23	/	-	**0.55**

/ indicates termination by lack of memory,
- indicates that the algorithm was still running after 2 days.

C. Comparison with Other Algorithms

We compared the performance of CPModule with four other methods, namely, Cister [9], Cluster-Buster [3], ModuleSearcher [4] and Compo [7]. We used the parameter values listed in table II and default values otherwise.

Table I shows the motif and nucleotide level CC scores for the different algorithms, when using all 516 transfac PWMs. Cister and Clusterbuster are able to find a solution, albeit of mediocre quality. The reason is that they operate on each sequence individually, which results in a rather large number of different motif predictions per sequence, most of which are false positives. ModuleSearcher was unable to finish because of memory problems, even when being allocated 2GB of ram. Compo was still running after 2 days and without any results returned. Compo also uses a strategy based on itemset mining, but does not address the problem of redundancy, which is inevitable when dealing with large numbers of motifs and hits.

To better compare the algorithms we run them with an increasing number of PWMs. We create different PWM sets by starting from the 3 PWMs of the inserted TFs, and sampling a number of additional PWMs from the set of 513 remaining PWMs. We independently sample 10 sets for every sample size. Figure 3 shows the correlation coefficient scores of the

TABLE II
NON-DEFAULT SETTINGS FOR ALTERNATIVE ALGORITHMS

Algorithm	Parameter	Setting
Cister	avg. distance between motifs	20
Cluster-Buster	gap	20
	residue abundance range	1000
Module-Searcher	search algorithm	genetic algorithm
	average number of motifs	6 (as Cister)
	maximum CRM size	165
	multiple copies of TF	no
	incomplete CRM penalty	no
	GA iterations	300
Compo	forbid overlaps	yes
	number of motifs	between 1 and 20
	distance window	165
	tp-factors	2, 3 and 4
	background sequences	same as CPModule

different algorithms when run with an increasing number of sampled PWMs. At the motif level, CPModule often preforms best, closely followed by ModuleSearcher. On the nucleotide level, however, CPModule performs significantly better than ModuleSearcher. Compo has a mostly constant performance at the motif level, however its nucleotide level score is rather low. Independent of the PWM set given, it returns just one of the inserted motifs on each of the sequences. We suspect that its screening method was too stringent, but found no way to change this behaviour. Cister and Cluster-Buster have low motif level scores, yet reasonably high nucleotide level scores, especially when using fewer PWMs. This shows that they are good tools for finding small CRMs on single sequences, but are not competitive to tools that operate on multiple sequences.

CPmodule on the other hand is able to effectively find CRMs over multiple sequences, even when being provided a large amount of PWMs. It is hence best in handling the combinatorial nature of CRM detection in *co*-regulated genes.

V. CONCLUSIONS

In this paper we presented how the problem of finding motif sets in co-regulated genes can be formulated in an existing constraint based itemset mining framework (CP4IM). This has several advantages: existing constraints such as frequency and their efficient implementation can be used out-of-the-box; the proximity constraint can easily be added to the system; existing itemset mining principles such as closedness can readily be applied. We demonstrated the usefulness of this approach by creating the CPModule system, based on CP4IM. This system can be used in a 3-phase method in which first a set of putative binding sites is computed by a motif detection tool. The CPModule system takes these binding sites as input and finds all sets of motifs that appear in each other's proximity in a sufficient number of sequences. This set of solutions is ranked according to p-value, calculated against a large set of background sequences. This method was evaluated experimentally: when we compare the output of the screening tool with the output of our method, we observe a significant reduction in false positives without significantly affecting true positives. The proximity constraint was shown to have a positive influence on predictive performance, while the sensitivity to this constraint was not too large. Compared to other tools, the predictive performance is usually competitive or better.

In future work we plan to further exploit the extensibility of our framework to deal with additional constraints: for instance, disallowing overlaps between motif hits or considering different strategies for avoiding statistical redundancies. Furthermore, we plan to apply our methodology on different types of data, such as ChIP-Seq data.

Acknowledgements 1) GOA AMBioRICS, GOA/08/011, CoE EF/05/007 SymBioSys 2) IUAP P6/25 3) Personal grant and project grant (SBO-BioFrame) from the agency for Innovation by Science and Technology in Flanders (IWT-Vlaanderen) 4) Postdoc grant from the Research Foundation—Flanders.

REFERENCES

[1] M. Tompa, N. Li, T. L. Bailey, G. M. Church, B. De Moor, E. Eskin, A. V. Favorov, M. C. Frith, Y. Fu, W. J. Kent, V. J. Makeev, A. A. Mironov, W. S. Noble, G. Pavesi, G. Pesole, M. Régnier, N. Simonis, S. Sinha, G. Thijs, J. van Helden, M. Vandenbogaert, Z. Weng, C. Workman, C. Ye, and Z. Zhu, "Assessing computational tools for the discovery of transcription factor binding sites," *Nature Biotechnology*, vol. 23, pp. 137–144, 2005.

[2] R. Sharan, I. Ovcharenko, A. Ben-Hur, and R. M. Karp, "Creme: a framework for identifying cis-regulatory modules in human-mouse conserved segments." *Bioinformatics*, vol. 19 (Suppl 1), 2003.

[3] M. C. Frith, M. C. Li, and Z. Weng, "Cluster-buster: finding dense clusters of motifs in DNA sequences," *Nucleic Acids Research*, vol. 31, no. 13, pp. 3666–3668, 2003.

[4] S. Aerts, P. Van Loo, Y. Moreau, and B. De Moor, "A genetic algorithm for the detection of new cis-regulatory modules in sets of coregulated genes." *Bioinformatics*, vol. 20, no. 12, pp. 1974–76, 2004.

[5] H. Sun, T. De Bie, V. Storms, Q. Fu, T. Dhollander, K. Lemmens, A. Verstuyf, B. De Moor, and K. Marchal, "ModuleDigger: an itemset mining framework for the detection of *cis*-regulatory modules," *BMC Bioinformatics*, vol. 10 (Suppl 1), 2009.

[6] A. Turi, C. Loglisci, E. Salvemini, G. Grillo, D. Malerba, and D. D'Elia, "Computational annotation of UTR cis-regulatory modules through frequent pattern mining," *BMC Bioinformatics*, vol. 10 (Suppl 6), 2009.

[7] G. Sandve, O. Abul, and F. Drabløs, "Compo: composite motif discovery using discrete models." *BMC bioinformatics*, vol. 9, no. 1, 2008.

[8] L. De Raedt, T. Guns, and S. Nijssen, "Constraint programming for itemset mining," in *KDD*, 2008, pp. 204–212.

[9] M. C. Frith, U. Hansen, and Z. Weng, "Detection of cis-element clusters in higher eukaryotic DNA," *Bioinformatics*, vol. 17, no. 10, pp. 878–889, 2001.

[10] M. C. Frith, Y. Fu, L. Yu, J.-F. Chen, U. Hansen, and Z. Weng, "Detection of functional DNA motifs via statistical over-representation," *Nucleic Acids Res*, vol. 32, no. 4, pp. 1372–81, 2004.

[11] P. Van Loo, S. Aerts, B. Thienpont, B. De Moor, Y. Moreau, and P. Marynen, "Moduleminer - improved computational detection of cis-regulatory modules: are there different modes of gene regulation in embryonic development and adult tissues?" *Genome biology*, vol. 9, no. 4, pp. R66+, 2008.

[12] A. Gallo, T. D. Bie, and N. Cristianini, "MINI: Mining informative non-redundant itemsets," in *PKDD*, 2007, pp. 438–445.

[13] Gecode Team, "Gecode: Generic constraint development environment," 2006, available from `http://www.gecode.org`.

[14] D. Xie, J. Cai, N.-Y. Chia, H. Ng, and S. Zhong, "Cross-species de novo identification of cis-regulatory modules with GibbsModule: application to gene regulation in embryonic stem cells," *Genome Research*, vol. 18, no. 8, pp. 1325–35, 2008.

[15] V. Matys, O. V. Kel-Margoulis, E. Fricke, I. Liebich, S. Land, A. Barre-Dirrie, I. Reuter, D. Chekmenev, M. Krull, K. Hornischer, N. Voss, P. Stegmaier, B. Lewicki-Potapov, H. Saxel, A. E. Kel, and E. Wingender, "TRANSFAC and its module TRANSCompel: transcriptional gene regulation in eukaryotes," *Nucleic Acids Research*, vol. 34 (Database Issue), pp. 108–110, 2006.

[16] B. Coessens, G. Thijs, S. Aerts, K. Marchal, F. De Smet, K. Engelen, P. Glenisson, Y. Moreau, J. Mathys, and B. De Moor, “INCLUSive: a web portal and service registry for microarray and regulatory sequence analysis,” *Nucleic Acids Research*, vol. 31, no. 13, pp. 3468–3470, 2003.

[17] K. Klepper, G. Sandve, O. Abul, J. Johansen, and F. Drabløs, “Assessment of composite motif discovery methods,” *BMC Bioinformatics*, vol. 9, 2008.