Approximating the maximum weighted decomposable graph problem with applications to probabilistic graphical models

Aritz Pérez APEREZ@BCAMATG.ORG
Basque Center for Applied Mathemtatics (BCAM)

Christian Blum CHRISTIAN.BLUM@IIIA.CSIC.ES
Artificial Intelligence Research Institute (IIIA-CSIC), Autonomous University of Barcelona (UAB)

Jose A. Lozano JA.LOZANO@EHU.EUS
University of the Basque Country (UPV/EHU); Basque Center for Applied Mathematics (BCAM)

Abstract

In this work we deal with the problem of learning a maximum weighted $(k+1)$-order decomposable graph coarser than a given maximal k-order decomposable graph (also known as hypertree of tree-width $k-1$). An Integer Linear Programming formulation for the problem has recently been proposed and used in order to solve instances of the problem with a moderate number of vertices. However, as the problem is known to be NP-hard, it is of practical interest to develop approximate algorithms able to work with a limited amount of computational resources. In this paper we propose an approximate Integer Linear Programming formulation for the problem using a threshold distance which discards the edges that, on average, have a low probability of being contained in the solution. Experiments have been carried out with weighted graphs and probabilistic graphical models. Using the proposed formulation we have obtained results close to the optimum, even when most of the candidate edges were discarded using the distance criterion. The obtained good results indicate that the approximate formulation has important applications for learning probabilistic graphical models using decomposable scores, e.g., BDe.

Keywords: Maximum weighted graph, decomposable graphs, bounded maximum clique size, decomposable models.

1. Introduction

The learning of probabilistic graphical models is usually divided in two parts: structural learning and parametric learning (Koller and Friedman, 2009). The most common approach to deal with the structural learning part consists in restricting the search space to structures with desirable properties. The structural learning can be faced using a search strategy guided by a score function that measures the goodness of the candidate graph. In some problems, the score function is additively decomposable in terms of structural components of the graph, such as, the edges or the complete set of vertices. The structural constraints plus the use of additively decomposable score functions facilitates the design of efficient learning algorithms. Sometimes, the structural learning is approached from the more general perspective of Graph Theory through a maximum weighted graph formulation of the problem. In maximum weighted graph problems the structures are constrained to the set of candidate structures of the original problem and the weight function is a generalization of the scoring function that inherits some of its useful properties, e.g. additively decomposable functions.

For instance, in Srebro (2000), the author formally defines the problem of learning the maximum weighted decomposable graphs with a maximum clique size of k from a graph theoretic perspective. We call these structures k-order decomposable graphs (kDG). The problem consists of finding a

kDG which maximizes the sum of the weights associated to its complete sets of vertices. The problem is NP-hard even for 0-1 weights associated to complete sets of size $k=3$, only. Recently, a set of exact algorithms have been proposed in order to deal with this problem in the context of graphical models (Corander et al., 2013; Kangas et al., 2014; Studený and Cussens, 2017). In Srebro (2000, 2003) an algorithm was proposed to deal with the maximum weighted kDG problem assuming that the weight function is monotone, i.e., the weight of a decomposable graph is equal or higher after the addition of an edge. Under this assumption the structures that solve the problem are known to be maximal k-order decomposable graphs (MkDG) (Malvestuto, 1991; Srebro, 2000). Since the log-likelihood is a monotone function and it is additively decomposable into the complete sets of vertices, the algorithm proposed by Srebro (2000, 2003) is applied to the problem of learning the maximum likelihood probabilistic model with kDG structure.

In this paper we deal with a maximum weighted graph problem closely related to the maximum weighted kDG problem: the maximum weighted $(k+1)$-order decomposable graph ($(k+1)$DG) problem (Pérez et al., 2016). This problem imposes an additional structural constraint which reduces the space of candidate graphs considered. The problem consists of learning a maximum weighted $(k+1)$DG coarser than a given MkDG (see Problem 1). In Pérez et al. (2016) is shown that the problem, despite of being a particular case of the problem proposed in (Srebro, 2000), is still an NP-hard problem. In Pérez et al. (2014) an equivalent formulation of the problem based on Integer Linear Programming (ILP) is proposed. This approach can be used as a building block for developing approximate algorithms in order to deal with the maximum weighted kDG problem (Pérez et al., 2016). Unfortunately, the number of decision variables and restrictions of the ILP formulation can grow cubically with the number of vertices of the graph, n (see Section 4). The poor scalability of the ILP formulation limits its applicability to problems of MkDGs with a moderate number of vertices. Moreover, since the problem is NP-hard, and in practice the available computational resources are limited, it is mandatory to develop computationally scalable approaches to this problem.

Based on a novel notion of distance in a graph between pairs of vertices, we propose an approximate ILP formulation for Problem 1 which discards a set of edges that, on average, has low probability of being included in the solution. The novel ILP formulation shows a better scalability with respect to n and, given a limited amount of computational resources, outperforms the original formulation even for moderate values of n. A set of experiments has been carried out concerning two types of structural learning problems: (1) maximum weighted graph and, (2) probabilistic graphical models with decomposable scores. Using the reduced formulation we have obtained results close to the optimum with a small fraction of the computational resources required by the general ILP formulation.

2. Maximum weighted $(k+1)$DG problem

Let $\boldsymbol{G}=(V,E)$ be an undirected graph, where $V=\{1,...,n\}$ is a set of indexes called the vertices and E is a set of pairs of vertices $\{u,v\}$ called edges. Let $\boldsymbol{G}^+=(V^+,E^+)$ be an undirected graph. $\boldsymbol{G}^+$ is **coarser** than $\boldsymbol{G}$ (or equivalently, $\boldsymbol{G}$ is **thinner** than $\boldsymbol{G}^+$) when $V=V^+$ and $E\subsetneq E^+$. An undirected graph is called a decomposable graph (DG) if for any cycle of length greater than 3 there exists a chord. Henceforth, we deal with DGs only. A set of vertices is said to be complete in $\boldsymbol{G}$ when it induces a complete subgraph. The set of all complete sets of vertices of $\boldsymbol{G}$ is denoted by $\mathbb{C}(\boldsymbol{G})$. The neighborhood of u in $\boldsymbol{G}$ is the set of vertices connected by an edge to u, $\mathbb{N}(u|\boldsymbol{G})=$

$\{v \in V : \{u, v\} \in E\}$. We define the neighborhood of a set of vertices S in $\boldsymbol{G}$ as the set of vertices connected by edges to all the vertices in S, $\mathbb{N}(S|\boldsymbol{G}) = \{v \in V : \forall u \in S, \{u, v\} \in E\} = \bigcap_{u \in S} \mathbb{N}(u|\boldsymbol{G})$. Note that this definition of neighborhood is the intersection of the neighborhood of each vertex instead of the union. A set $S \subseteq V \setminus \{u, v\}$ that separates u from v in $\boldsymbol{G}$ is called a separator for u and v. S is called minimal separator if any proper subset of S does not separate u and v in $\boldsymbol{G}$. From here on we will consider minimal separators only and we will call them separators, for the sake of brevity. The set of separators of $\boldsymbol{G}$ is denoted by $\mathbb{S}(\boldsymbol{G})$. A remarkable property of DGs is that their separators are complete sets (Lauritzen, 1996). We call to the number of separators for u and v in $\boldsymbol{G}$ the distance between u and v, and it is denoted as $l_{\boldsymbol{G}}(u, v)$. Note that the distance is 0 when $\{u, v\} \in E$ and the distance is 1 when u and v are at different connected components in $\boldsymbol{G}$. We say that an edge $\{u, v\} \notin E$ has length $l_G(\{u, v\})$ for G, that is, its length corresponds to the distance between u and v in $\boldsymbol{G}$. The components associated to a separator S is the set of sets of vertices that are mutually separated by S and it is denoted by $comp(S|\boldsymbol{G})$. Thus, given two different components $U, V \in comp(S|\boldsymbol{G})$, S is the minimal separator for any $u \in U$ and $v \in V$. We call to the size of $comp(S|\boldsymbol{G})$ the degree of the separator S, $d_S = |comp(S|\boldsymbol{G})|$, which corresponds to its multiplicity plus one.

Next, we show a theoretical characterization of the set of the edges that can be added to a DG maintaining its decomposability (Pérez et al., 2016). These edges are henceforth called **candidate edges**.

Theorem 1 *(Desphande et al., 2001) Given a DG $\boldsymbol{G} = (V, E)$, an edge $\{u, v\} \notin E$ is a **candidate edge** if and only if exists $S \subseteq V \setminus \{u, v\}$ such that (1) S is the separator for u and v in $\boldsymbol{G}$, and (2) $\{u, v\} \subset \mathbb{N}(S|\boldsymbol{G})$. We say that $\{u, v\}$ is a candidate edge due to S.*

Clearly, the addition of a candidate edge $\{u, v\}$ to $\boldsymbol{G}$ due to $S \in \mathbb{S}(\boldsymbol{G})$ creates the clique $\{u, v\} \cup S$. Besides, the addition of a set of edges can change the neighborhood of a separator S and, in consequence, the set of candidate edges due to S.

In this work we are particularly interested in two types of DGs which control explicitly the maximum clique size.

Definition 1 *A **k-order decomposable graph (kDG)** is a DG for which the maximum clique size is k. A **maximal k-order decomposable graph (MkDG)** is a kDG for which all the cliques are of size k and the addition of a candidate edge creates a clique of size $k + 1$.*

MkDGs are also known in the literature as $(k - 1)$-hypertrees (Srebro, 2000). We are interested in kDGs and MkDGs because Problem 1 is defined in terms of these types of DGs. An MkDG $\boldsymbol{G}$ has the following interesting structural properties (Malvestuto, 1991): (1) All the cliques of $\boldsymbol{G}$ are of size k, (2) $\boldsymbol{G}$ has $(k - 1)(2n - k)/2$ edges, and (3) the size of any separator $S \in \mathbb{S}(\boldsymbol{G})$ is $k - 1$. MkDGs are maximal in the sense that the addition of any candidate edge creates a clique of size greater than k. Henceforth, an MkDG will be denoted simply by $\boldsymbol{G}$ and a $(k + 1)$DG coarser than $\boldsymbol{G}$ (or $\boldsymbol{G}$ itself) will be denoted by $\boldsymbol{G}^+$.

The foundations being laid out, we can proceed to formally define the maximum weighted graph problem to be studied.

Problem 1 *Let $\boldsymbol{G}$ be an MkDG and let $\mathcal{W} = \{w_R : R \subseteq V, \text{ where } |R| \leq k+1\}$ be a set of weights associated to the sets of vertices of size lower than or equal to $k + 1$. Find a maximum weighted $(k + 1)$DG coarser than $\boldsymbol{G}$:*

$$\boldsymbol{G}^* = \arg_{\boldsymbol{G}^+ \in \mathcal{G}_{k+1}} \max w(\boldsymbol{G}^+) \tag{1}$$

where $\mathcal{G}_{k+1}$ is the set of $(k+1)$DGs coarser than $\boldsymbol{G}$ and $w(\boldsymbol{G}^+)$ is the weight function given by

$$w(\boldsymbol{G}^+) = \sum_{R \in \mathbb{C}(\boldsymbol{G}^+)} w_R \tag{2}$$

As we will see later, many of the weights included in $\mathcal{W}$ are irrelevant for solving Problem 1. However, in order to highlight the similarities with the more general problem (Srebro, 2000), we have decided to include the entire set $\mathcal{W}$ in the definition of Problem 1. Despite of being a special case, Problem 1 is still an NP-hard problem for $k \geq 1$ (Pérez et al., 2016). In this work we propose an approximate ILP formulation of this problem which (as will be shown later) obtains results very close to the optimum while making use of only a small fraction of the computational resources required by the general formulation.

According to Lemma 2.21 of (Lauritzen, 1996), any DG can be obtained starting from a thinner DG by adding candidate edges sequentially and, therefore, Problem 1 can be solved by considering the addition of candidate edges of successive DGs, only. Thus, in order to solve Problem 1 we are interested in the set of all candidate edges due to $S \in \mathbb{S}(\boldsymbol{G})$ that create cliques of size $k+1$ for an MkDG $\boldsymbol{G}$ or for any of the possibles $(k+1)$DGs coarser than $\boldsymbol{G}$, $\mathcal{G}_{k+1}$.

Definition 2 *Let $\boldsymbol{G}$ be an MkDG. An edge (u, v) is called **edge of interest** (EOI) due to S for $\boldsymbol{G}$ when it is a candidate edge for $\boldsymbol{G}$ or for a $(k+1)$DG $\boldsymbol{G}^+$ coarser than $\boldsymbol{G}$, and its addition to $\boldsymbol{G}$ and $\boldsymbol{G}^+$, respectively, creates the clique $\{u, v\} \cup S$ of size $k+1$. The EOIs due to S for $\boldsymbol{G}$ are denoted by $\mathbb{E}_S(\boldsymbol{G})$*

In order to characterize the set of cliques that can form part of the solution to Problem 1, we need to determine the set of separators of size $k-1$ for $\boldsymbol{G}$ and for any $\boldsymbol{G}^+$. Fortunately, this set corresponds to the set of separators for $\boldsymbol{G}$ (Pérez et al., 2016), that is, for any $\boldsymbol{G}^+$ coarser than $\boldsymbol{G}$ $\{S \in \mathbb{S}(\boldsymbol{G}) : |S| = k-1\} \subseteq \mathbb{S}(\boldsymbol{G})$. The set of EOIs for $\boldsymbol{G}$ is $\mathbb{E}(\boldsymbol{G}) = \bigcup_{S \in \mathbb{S}(\boldsymbol{G})} \mathbb{E}_S(\boldsymbol{G})$ where

$$\mathbb{E}_S(\boldsymbol{G}) = \{\{u, v\} : u \perp_{\boldsymbol{G}} v | S\}$$

is the set of EOIs due to S. By Definition 2 and Theorem 1, the EOI $\{u, v\} \in \mathbb{E}_S(\boldsymbol{G})$ is associated to the clique $\{u, v\} \cup S$.

Once we know the set of separators it is easy to see that the set of cliques of size $k+1$ of any $\boldsymbol{G}^+$ is given by $\{\{u, v\} \cup S : \{u, v\} \in \mathbb{E}_S(\boldsymbol{G}), \text{ for an } S \in \mathbb{S}(\boldsymbol{G})\}$. This result can reduce the set of cliques to be considered to solve Problem 1 from $\mathcal{O}(n^k)$ to $\mathcal{O}(n^3)$ and it allows to formulate Problem 1 as an ILP problem with $\mathcal{O}(n^3)$ decision variables, in the worst case.

By Theorem 1, the addition of the EOI $\{u, v\}$ due to S requires to add in advance the edges $\{\{u, s\} : s \in S\} \cup \{\{v, s\} : s \in S\}$, which generates $\boldsymbol{G}^+$. The addition of $\{u, v\}$ to $\boldsymbol{G}^+$ creates the new complete sets $\{\{u, v\} \cup R : R \subseteq S\}$, only, and therefore, the addition of $\{u, v\}$ to $\boldsymbol{G}^+$ creates the new complete sets $\{\{u, v\} \cup R : R \subseteq S\}$, only. Based on this observation it is possible to associate a weight to each EOI due to S.

Proposition 1 *Let $\boldsymbol{G}$ be an MkDG and let $\{u, v\}$ be an EOI due to S, $\{u, v\} \in \mathbb{E}_S(\boldsymbol{G})$. The contribution of $\{u, v\}$ due to S to the total weight of a solution is $w_{u,v|S} = \sum_{R \subseteq S} w_{\{u,v\} \cup R}$.*

At this point, the reader should note that in order to solve Problem 1 we only need the subset of $\mathcal{W}$ required to compute the weights associated to EOIs: $\mathcal{W}(\boldsymbol{G}) = \{w_{\{u,v\} \cup R} : R \subseteq S \text{ and } \{u, v\} \in \mathbb{E}_S(\boldsymbol{G}), \text{ where } S \in \mathbb{S}(\boldsymbol{G})\}$

3. An ILP formulation of Problem 1

This section presents a formulation based on Integer Linear Programing for solving Problem 1 (**exact formulation**, **EF**). Let $\boldsymbol{G} = (V, E)$ be the input MkDG. The **decision variables** of the EF are binary and correspond to the following set:

$$\{X_{u,v|S} : S \in \mathbb{S}(\boldsymbol{G}), \{u,v\} \in \mathbb{E}_S(\boldsymbol{G})\} \tag{3}$$

where

$$X_{u,v|S} = \begin{cases} 1, & \text{if the EOI } \{u,v\} \text{ due to } S \text{ is included in the solution.} \\ 0, & \text{if the EOI } \{u,v\} \text{ due to } S \text{ is NOT included in the solution.} \end{cases} \tag{4}$$

An edge $\{u,v\}$ is included in the solution if $X_{u,v|S} = 1$ for any $S \in \mathbb{S}(\boldsymbol{G})$, where $\{u,v\} \in \mathbb{E}_S(\boldsymbol{G})$, while $\{u,v\}$ is not included in the solution if $X_{u,v|S} = 0$ for all $S \in \mathbb{S}(\boldsymbol{G})$, where $\{u,v\} \in \mathbb{E}_S(\boldsymbol{G})$. The $(k+1)$DG encoded by the decision variables is $\boldsymbol{G}^+ = (V, E^+)$, where $E^+ = E \cup \{\{u,v\} : X_{u,v|S} = 1 \text{ for some } S \in \mathbb{S}(\boldsymbol{G})\}$. As we noted in Section 2, by Proposition 1 the addition of an EOI $\{u,v\}$ due to S contributes to the total weight of the obtained $(k+1)$DG with $w_{u,v|S}$ and, thus, the assignment $X_{u,v|S} = 1$ has the associated weight $w_{u,v|S}$.

Problem 1 can be stated in the following way in the form of an **ILP model**:

$$\max \sum_{S \in \mathbb{S}(\boldsymbol{G}), \{u,v\} \in \mathbb{E}_S(\boldsymbol{G})} w_{u,v|S} X_{u,v|S} \tag{5}$$

subject to the following set of **constraints**:

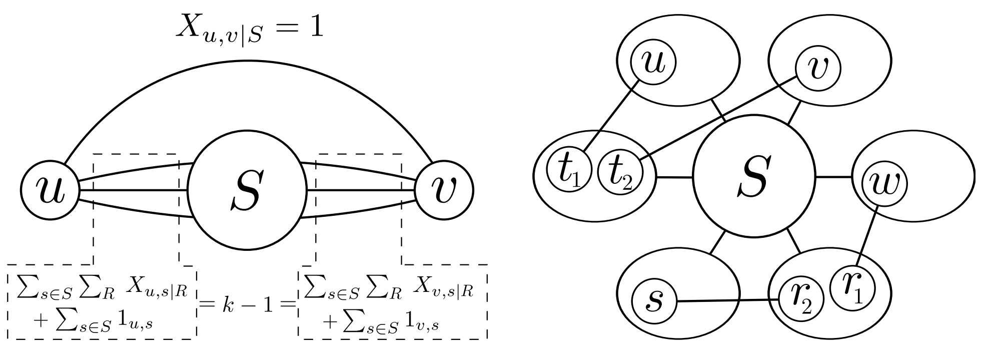
(1) $\sum_{S \in \mathbb{S}(\boldsymbol{G}) : \{u,v\} \in \mathbb{E}_S(\boldsymbol{G})} X_{u,v|S} \leq 1$ for $\{u,v\} \notin E$

(2) $[\sum_{R \in \mathbb{S}(\boldsymbol{G}) : \{u,v\} \in \mathbb{E}_R(\boldsymbol{G})} \sum_{s \in S} X_{u,s|R}] - (k-1) \cdot X_{u,v|S} \geq - \sum_{s \in S} 1_{u,s}$
$[\sum_{R \in \mathbb{S}(\boldsymbol{G}) : \{u,v\} \in \mathbb{E}_R(\boldsymbol{G})} \sum_{s \in S} X_{v,s|R}] - (k-1) \cdot X_{u,v|S} \geq - \sum_{s \in S} 1_{v,s}$
for $X_{u,v|S}$

(3) $\sum_{T \neq U \in \mathcal{C}} \sum_{u \in T} \sum_{v \in U} X_{u,v|S} \leq |\mathcal{C}| - 1$ for $S \in \mathbb{S}(\boldsymbol{G})$ and $\mathcal{C} \subset comp(S|\boldsymbol{G})$

The constants $1_{u,v}$ of constraints **(2)** take the value 1 if $\{u,v\} \in E$, and 0 otherwise. In constraints **(3)**, $\mathcal{C}$ represents a set of sets of vertices, and T and U are sets of vertices associated to (distinct) connected components of the graph induced by $V \setminus S$.

Constraints **(1)** guarantee that each EOI is considered by one separator, which ensures that each edge can be added only once in the solution. Constraints **(2)** guarantee that $\{u,v\}$ is in the neighborhood of S in the solution (see Theorem 1 and Figure 1(a)). In Section 2 we indicate that Problem 1 can be solved by adding sequentially a set of candidate edges (Lemma 2.21 in (Lauritzen, 1996)). The proposed ILP approach does not define an explicit ordering of the candidate edges added, but, it is possible to define the following partial ordering among the EOIs: if $\{u,v\}$ is an EOI due to S, then the edges $\{u,s\}$ and $\{v,s\}$ for $s \in S$ not included in $\boldsymbol{G}$ have to be added in advance. Constraints **(3)** guarantee that the vertices u and v were separated due to S before the edge was added. For example, in Figure 1(b), it is possible to add an edge from the component of v to the component of w because they are separated due to S. On the contrary, an edge from the component of v can not be added to the component of u because they are not separated due to S.


$X_{u,v|S} = 1$
u
S
v
$\sum_{s\in S}\sum_R X_{u,s|R} + \sum_{s\in S} 1_{u,s}$
$= k-1 =$
$\sum_{s\in S}\sum_R X_{v,s|R} + \sum_{s\in S} 1_{v,s}$
u
v
t_1
t_2
S
w
s
r_2
r_1

(a) Constraints **(2)**: These constraints ensure that if $\{u, v\}$ is added to the structure creating the clique $\{u, v\} \cup S$ then u and v are in the neighborhood of S.

(b) Constraints **(3)**: These constraints ensure that the edges between two vertices in different connected components form a forest.

Figure 1: This figure illustrates the constraints of the ILP approach to Problem 1. Small circles represent vertices, large circles separators, ellipses connected components and lines edges.

Note that constraints **(3)** guarantee that we add a forest among the connected components of each separator. At this point we would like to highlight that the proposed formulation guarantees that (1) all the EOIs can be found in at least a coarser $(k + 1)$DG and (2) each coarser $(k + 1)$DG is given by a unique configuration of the decision variables.

4. Size of the exact formulation

The bottleneck of the EF appears to be number of the constraints **(3)**, which is $\mathcal{O}(\sum_{S\in\mathbb{S}(\boldsymbol{G})} 2^{d_S})$. Note that this is exponential in d^{max}, the maximum degree of the separators of $\boldsymbol{G}$. The worst case corresponds to a MkDG with a single separator because in that case there are $\mathcal{O}(2^{n-k})$ constraints **(3)**. This fact limits the range of applications of the proposed approach to deal with MkDGs with moderate values of d^{max}. Fortunately, on average, the number of separators of degree d in a random tree (M2DG) decreases exponentially with d, $\mathcal{O}(\frac{n}{2^d})$. Thus, for $k = 2$, in the average case, it can be said that the number of constraints **(3)** grows linearly with n (see Section 6). This observation can be generalized to other values of k.

Surprisingly, in practice, the bottleneck of the EF lies in the number of decision variables and constraints **(1)** and **(2)** (see Table 1 and Figure 2(b)). The number of decision variables corresponds to the number of EOIs due to the different separators. In the worst case, this number is $\mathcal{O}(n^3)$. Note that $\{u, v\}$ with length l may be an EOI due to $l-1$ different separators. The number of constraints **(1)** can be $\mathcal{O}(n^2)$ because it is directly related with the number of different edges. The number of additive terms in the constraint associated to the edge $\{u, v\}$ corresponds to $l-1$. In the worst case, there are $\mathcal{O}(n^3)$ additive terms involved in constraint **(1)**. The number of constraints **(2)** corresponds to the number of EOIs due to the different separators which can be $\mathcal{O}(n^3)$. In addition, the number of terms involved in the constraint associated to the EOI $\{u, v\}$ (of length l) due to S_i is $\mathcal{O}(l)$. Thus, in the worst case the number of terms involved in constraints **(2)** is $\mathcal{O}(n^4)$. In summary, the number of decision variables can be $\mathcal{O}(n^3)$, and the number of constraints **(1)** and **(2)** amounts to $\mathcal{O}(n^3)$, with $\mathcal{O}(n^4)$ additive terms.

The number of variables and constraints involved in the EF cause ILP solvers to fail with growing problem size and under the assumption of a limited amount of computational resources. Moreover, even for rather small problems the necessary computation time for deriving an optimal solution might be unpractically high. In previous experiments (Pérez et al., 2014), it was observed a severe decrease in the quality of the obtained solutions (with a computation time limit of 3600 seconds) when the number of vertices approaches 100. The lack of scalability of the ILP model and the NP-hard nature of the problem motivates the proposal of an approximate ILP formulation which restricts the set of candidate solutions.

5. An approximated ILP formulation for Problem 1

In this section we present an alternative ILP formulation to deal with Problem 1 (**approximated formulation**, **AF**). This formulation imposes an additional structural constraint based on the length of the EOIs. The approach avoids the main difficulties of the EF and produces a reduced ILP problem more suitable for learning MkDGs with many vertices. Even though with this approach we can sacrifice the possibility to obtain optimal solutions for the original problem, we obtain high-quality approximate solutions within a limited amount of computation time (see Section 6).

The main idea of the formulation is to reduce the set of EOIs due to S by imposing a constraint based on a threshold distance l_{max}:

$$\mathbb{E}_S^{l_{max}}(\boldsymbol{G}) = \{\{u,v\} \in \mathbb{E}_S(\boldsymbol{G}) : l_{\boldsymbol{G}}(u,v) \leq l_{max}\}$$

where $1 < l_{max} < n$ is the threshold distance. In other words, an EOI $\{u,v\}$ due to S is only considered if there is a path whose length is at most l. The EF presented in Section 3 is modified by considering $\{u,v\}$ only if $l_{\boldsymbol{G}}(u,v) \leq l_{max}$. Note that this formulation generates (valid) solutions to Problem 1 because the constraints associated to an edge $\{u,v\}$ of length $l_{\boldsymbol{G}}(u,v)$ are given in terms of edges of lower length.

The motivations behind the use of a threshold distance l_{max} as a criterion to choose the EOIs to be considered by the AF is twofold: 1) Discard edges that, on average, have low probability of being added to the solution, and 2) reduce the size of the ILP model.

1. The addition of an EOI of length l implies the addition of $l-2$ EOIs of lower length being the largest of length at least $l/2$ and the smallest of length 2. Besides, the maximum number of edges that can be added to a MkDG to solve Problem 1 is $n-k$. Additionally, two different EOIs due to different separators may be incompatible due to their set of constraints **(2)** and **(3)**, and the possibilities to be incompatible increase with their length. As a consequence of these intuitions, it is clear that, generally, the number of EOIs of length l included decreases with l. Therefore, we can say that, the reduced formulation discards the EOIs that, on average (for random weight functions), are included in the optimal solution with low probability.

2. The number of decision variables increases polynomially with l_{max} and the increase in the number of variables implies an increase in the average number of additive terms of the constraints **(3)**. Moreover, the number of terms associated to an EOI of length l in constraints **(1)** and **(2)** is $\mathcal{O}(l_{\boldsymbol{G}}(u,v))$. Clearly, l_{max} can effectively control the number of EOIs and, in consequence, the number of decision variables, constraints **(1)** and **(2)**, and the number of additive terms in the constraints **(1)**, **(2)** and **(3)** of the AF (see Table 1).

In summary, as the distance of an EOI increases its addition becomes more unlikely, the rate of edges added with this distance is lower and the number of variables and constraints increases unnecessarily.

6. Experiments

The experimental results outlined in this section were obtained on a cluster of PCs with "Intel(R) Xeon(R) CPU E5450" CPUs of 8 nuclii of 3000 MHz and 32 Gigabyte of RAM. Moreover, the Integer Linear Programm (ILP) was solved with IBM ILOG CPLEX V12.1A run of CPLEX was stopped once (at least) 3600 CPU seconds had passed. The output (result) of CPLEX is the value of the best feasible integer solution found within the allowed CPU time limit. The results obtained by CPLEX are compared against the ones of a greedy approach where, at each step, the algorithm considers the addition of the candidate edge $\{u, v\}$ due to S with the maximum weight $w_{u,v|S}$ among the set of candidate edges.

Experimentation has been performed for two types of problems: (1) learning of maximum weighted $(k+1)$DGs and (2) learning of decomposable models with $(k+1)$DG structures using the Bayesian Dirichlet equivalent uniform score (Koller and Friedman, 2009). The experiments have been restricted to generating 3DGs from M2DGs (Problem 1 for $k = 2$). Further experimentation is left to future work.

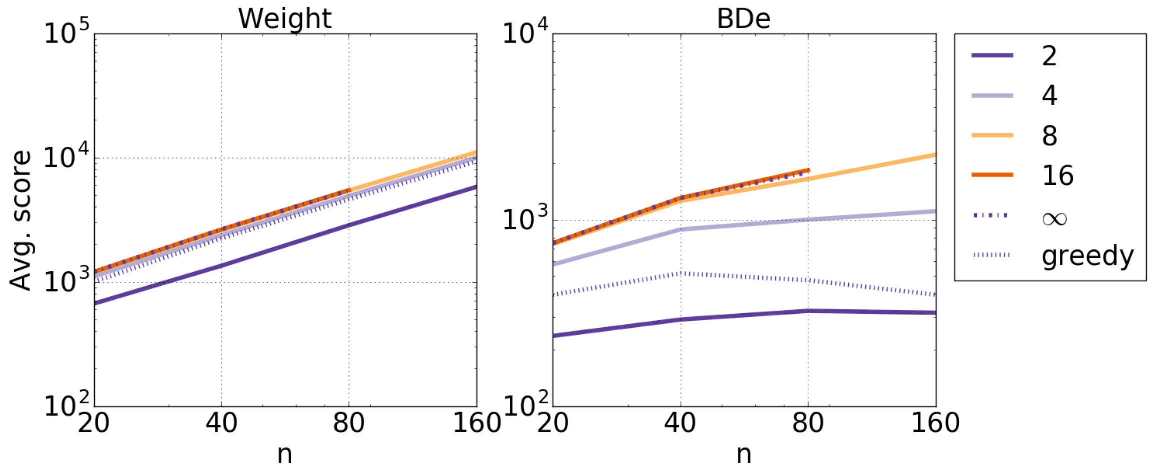
6.1 Maximum weighted $(k+1)$DG

The instances of Problem 1 have been randomly generated as follows: (1) A M2DG (a tree) with n vertices $\boldsymbol{G}$ is generated at random using the Kruskall algorithm for maximization where the weights associated to the edges are randomly generated. (2) The set of weights $\mathcal{W}(\boldsymbol{G})$ has been uniformly sampled from the set $\{-100, 99, ..., -1, 0, 1, ..., 99, 100\}$. 50 instances have been generated for each $n \in \{20, 40, 80, 160\}$. The instances with more than $50,000$ decision variables have been removed because the problem becomes inaccessible with 3600 seconds. The evolution of the obtained results with respect to n is illustrated in Figure 2. The summary of the obtained results is shown in Table 1.

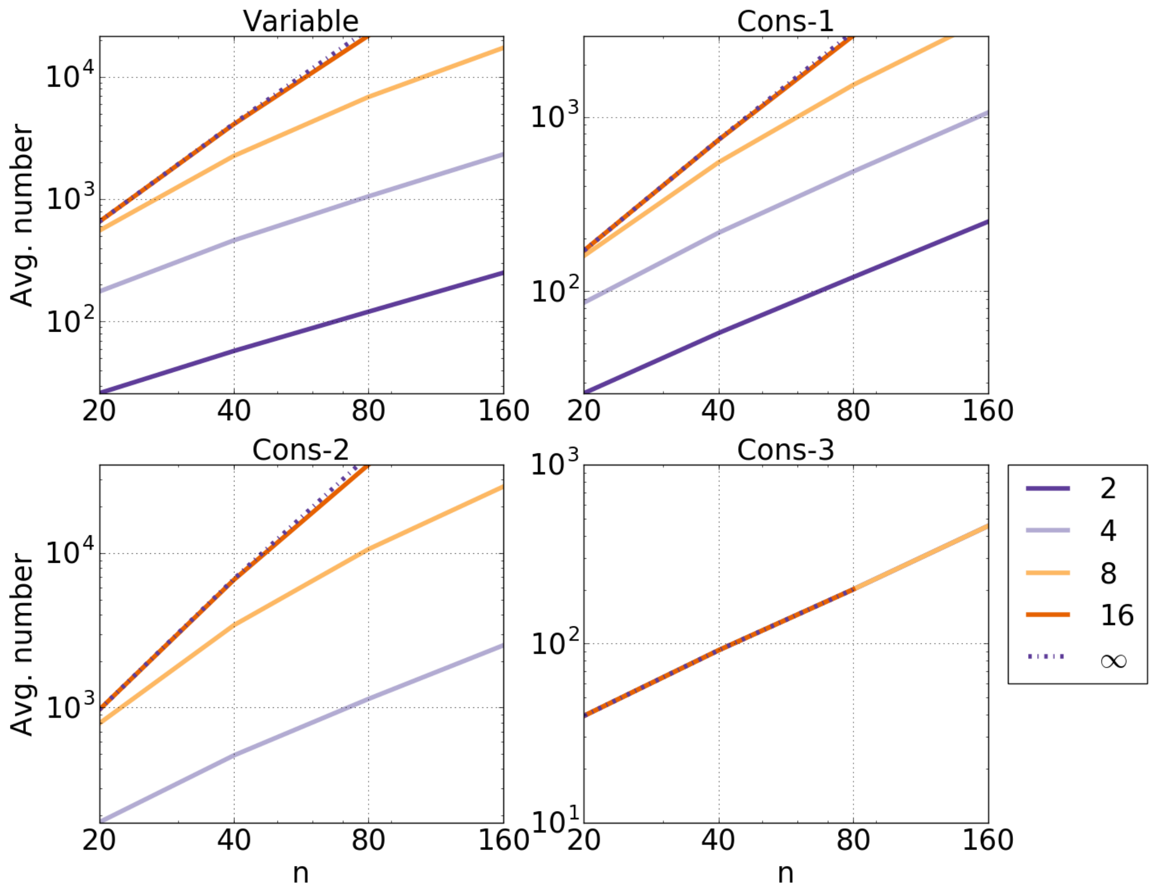
The results obtained for the AF with $l_{max} = 8$ are very similar to those of the EF, even for the instances for which the EF obtains the optimum value. Moreover, for $n = 80$, the AF with $l_{max} = 8$ obtains better results than the EF and, on average, it has a gap with respect to an upper bound of the optimum score close to 10%. The AF with l_{max} outperforms Greedy in all the instances, even when Greedy, on average, adds edges of length up to 6.18, 7.16 and 7.8 for $n = 40, 80, 160$, respectively. For $n = 80$ the AF with $l_{max} = 4$ has obtained a remarkable result, obtaining a solution close to the solution obtained with $l_{max} = 8$ and spending only an small fraction of the time (0,16 seconds). From the results it can be concluded that the best trade-off between the weight of the solution and the computational time is obtained with the order of thousands variables, being the computational time lower than 1 second and the quality of the solution close the best obtained (see for $n = 20, 40, 80, 160$ the AF with $l_{max} = 16, 8, 8, 4$, respectively).

6.2 Maximum BDe (k+1)DG

The instances of Problem 1 have been randomly generated as follows: (1) A M2DG with n vertices $\boldsymbol{G}$ is generated at random using the Kruskall algorithm for maximization where the weights asso-



(a) Weight



(b) Complexity

Figure 2: This figure visually shows the evolution of the average weights, and the average number of decision variables and contraints **(1)**, **(2)** and **(3)** with respect to the number of variables n.

Maximum weighted $(k+1)$DG										
	Problem features					Greedy	ILP			
n	l_{max}	Vars.	Cons. **(1)**	Cons. **(2)**	Cons. **(3)**	Weight	Weight	time	Gap	APD
	2	26.00	26.00	0.00			665.36	0.02	0.00	-34
	4	176.06	86.02	180.08			1098.94	0.02	0.00	11
20	8	552.98	159.58	786.80	39.40	996.60	1190.28	0.12	0.00	21
	16	653.64	171.00	965.28			1199.30	0.22	0.00	22
	∞	653.64	171.00	965.28			1199.30	0.23	0.00	22
	2	57.58	57.58	0.00			1342.68	0.02	0.00	-40
	4	461.00	216.48	489.04			2368.58	0.05	0.00	7
40	8	2255.92	548.68	3414.48	91.72	2231.06	2594.50	0.90	0.00	17
	16	4097.04	738.20	6717.68			2629.02	23.81	0.00	19
	∞	4143.14	1482.00	6804.28			2629.02	38.01	0.00	19
	2	120.46	120.46	0.00			2842.82	0.01	0.00	-38
	4	1054.26	486.02	1136.48			4894.48	0.08	0.00	7
80	8	6839.56	1526.08	10626.96	200.88	4586.68	5414.08	13.32	0.00	18
	16	21666.51	2923.66	37484.98			5443.20	2033.88	8.78	19
	∞	24470.53	3081.00	42779.06			5408.76	1839.53	10.69	18
	2	250.40	250.40	0.00			5785.18	0.01	0.00	-38
160	4	2321.34	1058.00	2526.68	454.16	9280.88	9959.82	0.16	0.00	7
	8	17210.62	3703.58	27014.08			10989.46	1204.90	0.44	19

Maximum BDe $(k+1)$DG										
	Problem features					Greedy	ILP			
n	l_{max}	Vars.	Cons. **(1)**	Cons. **(2)**	Cons. **(3)**	BDe	BDe	time	Gap	APD
	2	26.00	26.00	0.00			237.54	0.01	0.00	-39
	4	176.06	86.02	180.08			573.93	0.02	0.00	64
20	8	552.98	159.58	786.80	39.40	394.75	737.81	0.23	0.00	126
	16	653.64	171.00	965.28			747.12	0.36	0.00	130
	∞	653.64	171.00	965.28			747.12	0.40	0.00	130
	2	57.58	57.58	0.00			290.68	0.01	0.00	-40
	4	461.00	216.48	489.04			884.18	0.04	0.00	136
40	8	2255.92	548.68	3414.48	91.72	514.57	1257.55	3.85	0.00	269
	16	4097.04	738.20	6717.68			1304.75	330.92	0.97	284
	∞	4143.14	741.00	6804.28			1305.63	319.49	1.68	284
	2	120.46	120.46	0.00			323.62	0.01	0.00	-31
	4	1054.26	486.02	1136.48			998.26	0.04	0.00	147
80	8	6839.56	1526.08	10626.96	200.88	472.60	1649.52	168.36	0.17	333
	16	21694.60	2923.66	37541.88			1838.99	2924.95	42.72	385
	∞	24501.84	3081.00	42841.68			1789.32	2411.26	53.88	372
	2	250.40	250.40	0.00			316.26	0.01	0.00	-19
160	4	2321.34	1058.00	2526.68	454.16	395.17	1107.13	0.14	0.00	220
	8	17210.62	3703.58	27014.08			2225.06	779.55	4.80	592

Table 1: Summary of the results obtained for the generated Instances. The column l_{max} represents the threshold distance of the AF where the value ∞ represents the EF. The columns Vars., Cons-1, Cons-2 and Cons-3 show the average number of decision variables and constraints **(1)**, **(2)** and **(3)**, respectively, for the Integer Linear Programming formulation of the problem. The columns Weight (and BDe) show the average objective function values obtained by Greedy and ILP. The column Gap provides information about the average optimality gap (in percent), which refers to the gap between the value of the best valid solution that was found and the current lower bound at the time of stopping a run. The column APD is the average percentage deviation of ILP with respect to Greedy.

ciated to the edges are randomly generated. (2) Starting from a root vertex selected at random we have transformed $\boldsymbol{G}$ into a directed acyclic graph (DAG). (3) Directed edges are randomly added preserving until all the vertices has two parents in the resultant DAG $\boldsymbol{G}^{+}$. (4) A Bayesian networks $\boldsymbol{B}$ is constructed with the obtained structure $\boldsymbol{G}^{+}$ and its parameters have been uniformly sampled from a Dirichlet with the hyper-parameter equal to 0.5 (5) A data set $\mathcal{D}$ of size 500 has been sampled from the distribution represented by $\boldsymbol{B}$. (6) The set of weights have been calculated from $\mathcal{D}$ using the Bayesian Dirichlet equivalent uniform metric with an equivalent sample size of 1. 100 domains have been generated for each $n \in \{20, 40, 80, 160\}$. The instances with more than $50,000$ decision variables have been removed. We would like to highlight that, in general, the distribution represented by $\boldsymbol{B}$ can not be represented by a Markov network with a M3DG structure. Moreover, the treewidth of the obtained DAG is higher than 3 with a high probability and the optimum length for l_{max} can not be known in advance. This instances are more difficult because the percentage of EOIs with positives scores arranges from 30% for $n = 20$ to 5% for $n = 160$. The evolution of the obtained results with respect to n is illustrated in Figure 2. The summary of the obtained results is shown in Table 1.

The conclusions that can be drawn from these experiments are similar to the conclusions of the experiments summarized in Section 6.1. The best trade-off between the quality of the solutions and the required computational resources when the AF has the order of thousands decision variables ($l_{max} = 16, 8, 8, 4$ for $n = 20, 40, 80, 160$, respectively). In this case, the required time to find the optimum for the order of thousands variables grows up to hundred of seconds ($l_{max} = 8$ for $n = 80$). For $n = 160$ with the order of thousands variables the results are quite poor (see $l_{max} = 4$), however, we believe that setting an l_{max} between 5 and 7 we can obtain results with a good trade-off. In these experiments the APD obtained with respect to Greedy grows dramatically as n increases, even for $l_{max} = 4$.

7. Conclusions and future work

The proposed formulation transforms the ILP formulation of Problem 1 into an approximated ILP formulation of practical interest which is able to work with problems of learning maximum weighted graphs and decomposable models. The approximate formulation is based on a simple yet effective heuristic that reduces the number of candidate edges considered to solve Problem 1. By reducing the number of candidate edges, the proposed approximation can control the size of the produced ILP formulation (variables and constraints), which is the major bottleneck of the exact formulation (see Section 4). The heuristic is based on a threshold distance and it discard the edges that, on average, have a low probability of being contained in the optimal solution (see Section 5). The obtained experimental results confirm the intuitions behind the distance based constraint. The approximate formulation has obtained results close to the global optimum, even with small threshold distance values. For a limited amount of computational resources, e.g. 3600 seconds, we recommend to set the l_{max} parameter of the approximated formulation to a value which produces the order of thousands decision variables (see Subsection 6.1 and 6.2 for further details). The proposed approximated ILP formulation has shown to be an effective strategy in order to approach Problem 1.

In the future, we will develop strategies to further reduce the number of edges to be considered by the approximated ILP formulation taking into account the set of weights of a particular instance of Problem 1. Additionally, we will design efficient algorithms to deal with the more general maximum weighted K-order decomposable graph problem defined by Srebro (2000). This algorithms will be based on the divide-and-conquer strategy presented by Pérez et al. (2016), which construct a sequence of coarser MiDGs for $i = 2, ..., k$. The divide-and-conquer approach has shown to be an effective strategy for learning decomposable models with a bounded maximum clique size.

ACKNOWLEDGMENTS.

This work has been partially supported by the Basque Government through the BERC 2014-2017 and the ELKARTEK programs, and by the Spanish Ministry of Economy and Competitiveness MINECO: BCAM Severo Ochoa excellence accreditation SVP-2014-068574 and SEV-2013-0323, and through the project TIN2017-82626-R funded by (AEI/FEDER, UE).

References

J. Corander, T. Janhunen, J. Rintanen, H. Nyman, and J. Pensar. Learning chordal Markov networks by constraint satisfaction. In *Advances in Neural Information Processing Systems*, pages 1349–1357, 2013.

A. Desphande, M. Garofalakis, and M. I. Jordan. Efficient stepwise selection in decomposable models. In *Uncertainty and Artificial Intelligence*, pages 128–135, 2001.

K. Kangas, T. Niinimäki, and M. Koivisto. Learning chordal Markov networks by dynamic programming. In *Advances in Neural Information Processing Systems*, pages 2357–2365, 2014.

D. Koller and N. Friedman. *Probabilistic Graphical Models. Principles and Techniques*. The MIT Press, Cambridge, Massachusetts, 2009.

S. L. Lauritzen. *Graphical Models*. Oxford University Press, New York, 1996.

F. M. Malvestuto. Approximating discrete probability distributions with decomposable models. *IEEE Transactions on Systems, Man and Cybernetics*, 21(5):1287–1294, 1991.

A. Pérez, C. Blum, and J. A. Lozano. Learning maximum weighted (k+1)-order decomposable graphs by Integer Linear Programming. In *Probabilistic Graphical Models (PGM). Lecture Notes in Computer Science*, volume 8754, pages 396–408. Springer, Cham, 2014.

A. Pérez, I. Inza, and J. A. Lozano. Efficient approximation of probability distributions with k-order decomposable models. *Journal of Approximate Reasoning*, 74:58–97, 2016.

N. Srebro. Maximum likelihood Markov networks: An algorithmic approach. Master thesis, Massachuset Institute of Technology, 2000.

N. Srebro. Maximum likelihood bounded tree-width Markov networks. *Artificial Intelligence*, 143: 123–138, 2003.

M. Studený and J. Cussens. Towards using the chordal graph polytope in learning decomposable models. *International Journal of Approximate Reasoning*, 88:259–281, 2017.