# AIGM'12

**ECAI workshop on algorithmic issues for inference in graphical models**

27th/28th of August 2012

Montpellier, France

## Motivation

Most real (e.g. biological) complex systems are formed or modelled by elementary objects that locally interact with each other. Local properties can often be measured, assessed or partially observed. On the other hand, global properties that stem from these local interactions are difficult to comprehend. It is now acknowledged that a mathematical modelling is an adequate framework to understand, be able to control or to predict the behaviour of complex systems, such as gene regulatory networks or contact networks in epidemiology. More precisely, graphical models (GM), which are formed by variables bound to their interactors by deterministic or stochastic relationships, allow researchers to model possibly high-dimensional heterogeneous data and to capture uncertainty. Analysis, optimal control, inference or prediction about complex systems benefit from the formalisation proposed by GM. To achieve such tasks, a key factor is to be able to answer general queries: what is the probability to observe such event in this situation ? Which model best represents my data ? What is the most acceptable solution to a query of interest that satisfies a list of given constraints ? Often, an exact resolution cannot be achieved either because of computational limits, or because of the intractability of the problem.

## Objective

The aim of this workshop is to bridge the gap between Statistics and Artificial Intelligence communities where approximate inference methods for GM are developped. We are primarily interested in algorithmic aspects of probabilistic (e.g. Markov random fields, Bayesian networks, influence diagrams), deterministic (e.g. Constraint Satisfaction Problems, SAT, weighted variants, Generalized Additive Independence models) or hybrid (e.g. Markov logic networks) models. We expect both (i) reviews that analyze similarities and differences betwen approaches developped by computer scientists and statisticians in these areas, and (ii) original papers that propose new algorithms and show their performance on data sets as compared to state-of-the-art methods.

## For whom

Researchers from the fields of Machine Learning, Constraint satisfaction and optimisation, decision theory, Probability and Statistics

The workshop organizers
Nathalie Peyrard (BIA, INRA Toulouse, France)
Stéphane Robin (AgroParisTech, Paris, France)
Thomas Schiex (BIA, INRA Toulouse, France)
would like to thank the following sponsors

- Ficolofo, (ANR project),
- The CNRS GDR Santé,
- MSTGA, (INRA, MIA internal research network)
- MIA (INRA Applied maths and Computer science dept.)

## Provisional Program

- 9h15 Workshop introduction

- 9h30 – 10h30 Invited talk: J. Larrosa and E. Rollon. Algorithms for Graphical Models: Mapping the global picture.

- 10h30 – 11h00 break

- 11h00 – 11h30 D. Allouche, C. Bessiere, P. Boizumault, S. de Givry, P. Gutierrez, S. Loudni, JP. Metivier, T. Schiex. Filtering Decomposable Global Cost Functions
- 11h30 - 12h00 J. Larrosa and E. Rollon. Improved Bounded Max-Sum for Distributed Constraint Optimization
- 12h00 – 12h30 W. Probert, E. McDonald-Madden, N. Peyrard, R. Sabbadin. Computational issues surrounding the dynamic optimisation of management of an ecological food web

- 12h30 – 14h00 lunch break

- 14h00 – 15h00 Invited talk: A. Globerson. Linear Programming Relaxations for Graphical Models - Learning and Inference
- 15h00 – 15h30 J. Cussens.An upper bound for BDeu local scores

- 15h30 - 15h45 break

- 15h45 – 16h15 J. Vandel, B. Mangin, S. de Givry. New local move operators for learning the structure of Bayesian networks
- 16h15 – 16h45 P. Zhang, F. Krzakala, J. Reichardt and L. Zdeborova. Comparative Study for Inference of Hidden Classes in Stochastic Block Models

## Invited talks

Javier Larrosa and Emma Rollon:
**Algorithms for Graphical Models: Mapping the global picture.**

Abstract: several fields of research can be cast into the general framework of Graphical Models. Independent research in the different fields have often developed similar ideas. However, this similarity is often hard to realize because of each field semantics, notation, etc. The purpose of this tutorial is to describe the main common algorithmic techniques in the most general terms. We expect that it will facilitate researchers from different specific fields to realize where "their" techniques fit and how to benefit from similar ideas from different fields. We will do our best to describe general algorithms using standard text-books language (e.g. dynamic programming, memoization, greedy algorithms, relaxation, etc) and avoiding domain-specific one (e.g. inference, consistency enforcement, case-based reasoning, etc). More specific algorithms, mainly from Weighted CSPs and Max-SAT, will subsequently described.

Amir Globerson
**Linear Programming Relaxations for Graphical Models - Learning and Inference**

Abstract: Many probabilistic modeling tasks rely on solving challenging inference problems. These combinatorial problems arise, e.g., in predicting likely values for variables as in selecting and orienting residues in protein design, parsing in natural language processing, or when learning the model structure itself. In many cases, the inference problems involve densely connected variables (or higher order dependences) and are provably hard. However, recent research has shown that some of these difficulties can be overcome by a careful choice of approximation schemes and learning algorithms. These have yielded very encouraging results in wide array of fields, from machine vision and natural language processing to computational biology and signal processing. The tutorial will focus on linear programming (LP) relaxations which have been particularly successful in solving inference problems. Intuitively, LP relaxations decompose a complex problem into a set of simpler subproblems that are subsequently encouraged to agree. If the subproblems agree about a solution, then the solution is the optimal one, otherwise it is fractional. Geometrically, the relaxation maintains an outer bound approximation to a polytope whose vertexes correspond to valid solutions. I will introduce and explain key ideas behind recent approaches, discuss when they can and cannot be applied, how they can be integrated into supervised learning schemes and what efficient message passing algorithms exist for solving them. Examples from several applications will be provided, including computational biology, natural language processing, and structure learning.

# Filtering Decomposable Global Cost Functions [1]

**D. Allouche[1] C. Bessiere[2] P. Boizumault[3] S. de Givry[1] P. Gutierrez[4] S. Loudni[3] JP. Métivier[3] T. Schiex[1]**
[1]**UBIA UR 875, INRA, F-31320 Castanet Tolosan, France**
[2]**U. Montpellier, France**
[3]**GREYC-CNRS, UMR 6072, U. Caen, France**
[4]**IIIA-CSIC, U. Autònoma de Barcelona, Bellaterra, Spain**

**Abstract.**

As [18] have shown, weighted constraint satisfaction problems can benefit from the introduction of global cost functions, leading to a new Cost Function Programming paradigm. In this paper, we explore the possibility of decomposing global cost functions in such a way that enforcing soft local consistencies on the decomposition achieves the same level of consistency on the original global cost function. We give conditions under which directional and virtual arc consistency offer such guarantees. We conclude by experiments on decomposable cost functions showing that decompositions may be very useful to easily integrate efficient global cost functions in solvers.

## Introduction

Graphical model processing is a central problem in artificial intelligence. The optimization of the combined cost of local cost functions, central in the valued/weighted constraint satisfaction problem frameworks [24] federates a variety of famous problems including CSP, SAT, Max-SAT, but also the *Maximum A posteriori Problem* (MAP) in Random Markov fields, the *Maximum Probability Explanation* (MPE) problem in Bayes nets [14] and polynomial pseudo-Boolean optimization [6]. It has applications in resource allocation or bioinformatics.

The main approach to solve such problems in the most general situation relies on Branch and Bound combined with dedicated lower bounds for pruning. Such lower bounds can be provided by enforcing soft local consistencies [7], as in Constraint Programming (CP) solvers. CP solvers are also equipped with global constraints which are crucial for solving large difficult problems. Dedicated algorithms for filtering such constraints have been introduced. For some global constraints such as REGULAR, CONTIGUITY, AMONG, it has been shown that a decomposition into a Berge-acyclic network of fixed arity constraints can lead to simpler implementation, without any loss in efficiency or effectiveness in filtering [2, 4].

The notion of global constraints has been recently extended to weighted CSP, defining Global Cost Functions [27, 18] with associated efficient filtering algorithms. In this paper, after some preliminaries, we define cost function decomposition and show how decomposable global constraints can be softened in families of decomposable global cost functions with the same decomposition structure. For Berge-acyclic decomposable global cost functions, we show that enforcing directional arc consistency or virtual arc consistency on the decomposition is essentially equivalent to a direct application on the original global cost function. Finally, we experimentally compare the efficiency of decomposed and monolithic versions of different global cost functions and observe important speedups using decompositions.

[1] This work has been funded by the "Agence nationale de la Recherche", reference ANR-10-BLA-0214.

## 1 Preliminaries

### 1.1 Cost function network.

A Cost Function Network (CFN) is a pair $(X, W)$ where $X = \{1, \ldots, n\}$ is a set of $n$ variables and $W$ is a set of cost functions. Each variable $i \in X$ has a finite domain $D_i$ of values that can be assigned to it. A value $a$ in $D_i$ is denoted $(i, a)$. The maximum domain size is $d$. For a set of variables $S \subseteq X$, $D^S$ denotes the Cartesian product of the domains of the variables in $S$. For a given tuple of values $t$, $t[S]$ denotes the projection of $t$ over $S$. A cost function $w_S \in W$, with scope $S \subseteq X$, is a function $w_S : D^S \mapsto [0, k]$ where $k$ is a maximum integer cost (finite or not) used to represent forbidden assignments (expressing hard constraints). To faithfully capture hard constraints, costs are combined using the bounded addition defined by $\alpha \oplus \beta = \min(k, \alpha + \beta)$. In this paper, a hard constraint is therefore represented as a cost function using only costs in $\{0, k\}$. If $\forall t \in D^S, z_S(t) \leq w_S(t)$, we say that the cost function $z_S$ is a relaxation of $w_S$, denoted by $z_S \leq w_S$. A cost $\beta$ may be subtracted from a larger cost $\alpha$ using the operation $\ominus$ where $\alpha \ominus \beta$ is $(\alpha - \beta)$ if $\alpha \neq k$ and $k$ otherwise. Without loss of generality, we assume that every network contains one unary cost function $w_i$ per variable and a 0-arity (constant) cost function $w_\varnothing$.

The central problem in CFN is to find an optimal *solution*: a complete assignment $t$ minimizing the combined cost function $\bigoplus_{w_S \in W} w_S(t[S])$. This optimization problem has an associated NP-complete decision problem and restrictions to Boolean variables and binary constraints are known to be APX-hard [20].

A Constraint Network (CN) is a CFN where all cost functions are hard constraints (i.e., only using costs in $\{0, k\}$). Such cost functions are simply called constraints.

### 1.2 Local consistency.

Algorithms searching for solutions in CNs usually enforce local consistency properties to reduce the search space. In CNs, the standard level of local consistency is generalized arc consistency (GAC). A

constraint $c_S$ is GAC iff every value in the domain of every variable in $S$ has a support on $c_S$, where a support on $c_S$ is a tuple $t \in D^S$ such that $c_S(t) = 0$. Enforcing GAC on $c_S$ will often be called *filtering* $c_S$. General exact methods for solving the minimization problem in CFNs usually rely on branch and bound algorithms equipped with dedicated lower bounds. We consider here the incremental lower bounds provided by maintaining soft local consistencies such as directed arc consistency (DAC) [8, 17] and virtual arc consistency (VAC) [7].

### 1.3 Global cost function.

A global constraint $c(S, \theta)$ is a family of constraints with a precise semantics parameterized by the set of variables $S$ involved and possible extra parameters represented as $\theta$. Global constraints usually have efficient associated local consistency enforcing algorithm (compared to generic filtering algorithms). Global constraints have been extended to define soft global constraints such as SOFTALLDIFF$(S)$ [22] or SOFTREGULAR$(S, \mathcal{A}, d)$ [26]).

These "soft" global constraints are in fact hard global constraints including one auxiliary variable in their scope representing the amount of violation of the assignment of the original variables. This amount of violation depends on the semantics of violation used for the softening of that global constraint. For several such constraints, efficient dedicated algorithms for enforcing GAC have been proposed.

Recently, different papers [27, 18] have shown that it is possible to define soft global constraints as parameterized cost functions $z(S, \theta)$ directly providing the cost of an assignment. This approach allows to directly enforce soft local consistencies with dedicated algorithms providing stronger lower bounds. Indeed, compared to the previous cost variable based approach using constraints and GAC, cost functions and soft local consistencies offer improved filtering, thanks to the enhanced communication between cost functions enabled by the use of Equivalence Preserving Transformations [9].

### 1.4 Hypergraph.

The hypergraph of a CFN (or CN) $(X, W)$ has one vertex per variable $i \in X$ and one hyperedge per scope $S$ such that $\exists w_S \in W$. We consider CFNs with connected hypergraphs. The incidence graph of an hypergraph $(X, H)$ is a graph $G = (X \cup H, H')$ where $\{x_i, e_j\} \in H'$ iff $x_i \in X, e_j \in H$ and $x_i$ belongs to the hyperedge $e_j$. An hypergraph $(X, H)$ is Berge acyclic iff its incidence graph is acyclic.

## 2 Decomposing Global Cost Functions

Some global constraints may be efficiently decomposed into a logically equivalent subnetwork of constraints of bounded arities [5, 3]. Similarly, global cost functions may be decomposed into a set of bounded arity cost functions. Notice that the definition below applies to any cost function, including constraints (cost functions using only costs in $\{0, k\}$).

**Definition 1** *A decomposition of a global cost function $z(T, \theta)$ is a polynomial transformation $\delta_p$ ($p$ being an integer that bounds arity) that returns a CFN $\delta_p(T, \theta) = (T \cup E, F)$ such that $\forall w_S \in F, |S| \leq p$ and $\forall t \in D^T, z(T, \theta)(t) = \min_{t' \in D^{T \cup E}, t'[T]=t} \bigoplus_{w_S \in F} w_S(t'[S])$.*

We assume, w.l.o.g, that every auxiliary variable $i \in E$ is involved in at least two cost functions in the decomposition.[2] Clearly, if $z(T, \theta)$ appears in a CFN $P = (X, W)$ and decomposes into $(T \cup E, F)$, then the optimal solutions of $P$ can directly be obtained by projecting the optimal solutions of the CFN $P' = (X \cup E, W \setminus \{z(T, \theta)\} \cup F)$ on $X$.

**Example** Consider the ALLDIFF$(S)$ constraint and its associated softened variant SOFTALLDIFF$(S, dec)$ using the *decomposition* measure [22] where the cost of an assignment is the number of pairs of variables taking the same value. It is well known that ALLDIFF decomposes in a set of $\frac{n.(n-1)}{2}$ binary difference constraints. Similarly, the SOFTALLDIFF$(S, dec)$ cost functioncan be decomposed in a set of $\frac{n.(n-1)}{2}$ soft difference cost functions. A soft difference cost function takes cost 1 iff the two involved variables have the same value and 0 otherwise. In these cases, no auxiliary variable is required. Notice that the two decompositions have the same hypergraph structure.

### 2.1 Softening Decomposable Global Constraints

We now show that there is a systematic way of deriving decomposable cost functions as specific relaxations of existing decomposable global constraints.

As the previous ALLDIFF example showed, if we consider a decomposable global constraint, it is possible to define a softened decomposable global cost function by relaxing every constraint in the decomposition.

**Theorem 1** *Let $c(T, \theta)$ be a global constraint that decomposes in a constraint network $(T \cup E, C)$ and $f_\theta$ a function that maps every $c_S \in C$ to a cost function $w_S$ such that $w_S \leq c_S$. Then the global cost function $w(T, f_\theta)(t) = \min_{t' \in D^{T \cup E}, t'[T]=t} \bigoplus_{c_S \in C} f_\theta(c_S)(t'[S])$ is a relaxation of $c(T, \theta)$.*

**Proof** For any tuple $t \in D^T$, if $c(T, \theta)(t) = 0$, then $\min_{t' \in D^{T \cup E}, t'[T]=t} \bigoplus_{c_S \in C} c_S(t'[S]) = 0$ because $(T \cup E, C)$ is a decomposition of $c(T, \theta)$. Let $t' \in D^{T \cup E}$ be the tuple where this minimum is reached. This implies that $\forall c_S \in C$, $c_S(t'[S]) = 0$. Since $f_\theta(c_S)$ is a relaxation of $c_S$, this implies that $f_\theta(c_S)(t'[S]) = 0$ too. Therefore $\bigoplus_{c_S \in C} f_\theta(c_S)(t'[S]) = 0$ and $w(T, f_\theta)(t) = 0$. □

By definition, the global cost function $w(T, f_\theta)$ is decomposable in $(T \cup E, W)$ where $W$ is obtained by mapping $f_\theta$ on every element of $C$. Notice that, since $f_\theta$ preserves scopes, the hypergraph of the decomposition is preserved.

This result allows to immediately derive a long list of decompositions for global cost functions from existing decompositions of global constraints such as ALLDIFF, REGULAR, GRAMMAR, AMONG, STRETCH. The parameterization through $f_\theta$ allows a lot of flexibility.

Consider the ALLDIFF$(V)$ constraint decomposed into a clique of binary differences. From a graph $G = (V, H)$, one can define a relaxation function $f_G$ that preserves difference constraints $i \neq j$ when $(i, j) \in H$ but otherwise relaxes them to a constant cost function that is always equal to zero. This gives rise to a global cost function $w(V, f_G)$ that captures the graph coloring problem on $G$, an

[2] Otherwise, such a variable can be removed by variable elimination: remove $i$ from $E$ and replace the $w_S$ involving $i$ by the cost function $\min_i w_S$ on $S \setminus \{i\}$. This preserves Berge-acyclicity.

NP-hard problem. Thus, enforcing DAC or VAC on that single global cost function will be intractable as well, whereas enforcing DAC or VAC on its decomposition into binary cost functions will obviously be polynomial but will hinder the level of filtering achieved.

Consider the REGULAR$(\{X_1, \ldots, X_n\}, \mathcal{A})$ global constraint, defined by a finite automaton $\mathcal{A} = (Q, \Sigma, \delta, q_0, F)$ where $Q$ is a set of states, $\Sigma$ the emission alphabet, $\delta$ a transition function from $\Sigma \times Q \rightarrow 2^Q$, $q_0$ the initial state and $F$ the set of final states. As shown in [4], this constraint decomposes into a constraint network $(\{X_1, \ldots, X_n\} \cup \{Q_0, \ldots, Q_n\}, C)$ where the auxiliary variables $Q_i$ have $Q$ as their domain. The set of constraints $C$ in the decomposition contains two unary constraints restricting $Q_0$ to $\{q_0\}$ and $Q_n$ to $F$ and a sequence of identical ternary constraints $c_{\{Q_i, X_{i+1}, Q_{i+1}\}}$ which allows a triple $(q, s, q')$ iff $q' \in \delta(q, s)$, thus capturing $\delta$. An arbitrary relaxation of this decomposition may relax each of these constraints. The unary constraints on $Q_0$ and $Q_n$ would be replaced by unary cost functions $\lambda_{Q_0}$ and $\rho_{Q_n}$ stating the cost for using every state as either an initial or final state while the ternary constraints would be relaxed to ternary cost functions $\sigma_{\{Q_i, X_{i+1}, Q_{i+1}\}}$ stating the cost for using any $(q, s, q')$ transition. This relaxation precisely corresponds to the use of a weighted automaton $\mathcal{A} = (Q, \Sigma, \lambda, \sigma, \rho)$ [11]. The cost of an assignment in the decomposition is equal, by definition, to the cost of an optimal parse of the assignment by the weighted automaton. This defines a WEIGHTEDREGULAR$(\{X_1, \ldots, X_n\}, \mathcal{A})$ global cost function. As shown in [13], a weighted automaton can encode the Hamming and Edit distances to the language of a classical automaton. Contrary to the ALLDIFF example, we will see that WEIGHTEDREGULAR decomposition can be handled efficiently and effectively by soft local consistencies.

## 3 Local Consistency and Decompositions

The use of decompositions instead of their monolithic variant has both advantages and drawbacks. Thanks to local reasoning, a decomposition may be filtered more efficiently but this may also hinder the level of filtering achieved. In classical CSP, it is known that if the decomposition is Berge-acyclic, then enforcing GAC on the decomposition enforces GAC on the global constraint itself [1]. We show that a similar result can be obtained for cost functions using either DAC or VAC.

DAC has been originally introduced on binary cost functions using the notion of full support [7]. For a cost function $w_S$, a tuple $t \in D^S$ is a full support for a value $(i, a)$ of $i \in S$ iff $w_i(a) = w_S(t) \bigoplus_{j \in S} w_j(t[j])$. Notice that either $w_i(a) = k$ and $(i, a)$ does not participate in any solution or $w_i(a) < k$ and therefore $w_S(t) \bigoplus_{j \in S, j \neq i} w_j(t[j]) = 0$. DAC has been extended to non binary cost functions in [23] and [19] with different definitions that coincide on binary cost functions. In this paper, we use a simple extension called T-DAC (for terminal DAC). Given a total order $\prec$ on variables, a CFN is said to be T-DAC w.r.t. $\prec$ iff for any cost function $w_S$, any value $(i, a)$ of the maximum variable $i \in S$ according to $\prec$ has a full support on $w_S$.

VAC is a more recent local consistency property that establishes a link between a CFN $P = (X, W)$ and a constraint network denoted as $Bool(P)$ with the same set $X$ of domain variables and which contains, for every cost function $w_S \in W, |S| > 0$, a constraint $c_S$ with the same scope which forbids any tuple $t \in D^S$ such that $w_S(t) \neq 0$. A CFN $P$ is said to be VAC iff the arc consistent closure of the constraint network $Bool(P)$ is non empty [7].

### 3.1 Enforcing soft local consistencies

Enforcing such soft local consistencies relies on arc level *Equivalence Preserving Transformations* (EPTs) which apply to one cost function $w_S$ [9]. Instead of deleting domain values, EPTs shift costs between $w_S$ and the unary constraints $w_i, i \in S$ and therefore operate on a sub-network of $P$ defined by $w_S$ and denoted as $N_P(w_S) = (S, \{w_S\} \cup \{w_i\}_{i \in S})$. The main EPT is described as Algorithm 1. This EPT shifts an amount of cost $|\alpha|$ between the unary cost function $w_i$ and the cost function $w_S$. The direction of the cost move is given by the sign of $\alpha$. The precondition guarantees that costs remain non negative in the resulting equivalent network.

**Algorithm 1**: A cost shifting EPT used to enforce soft arc consistencies. The $\oplus, \ominus$ operations are extended to handle possibly negative costs as follows: for non negative costs $\alpha, \beta$, we have $\alpha \ominus (-\beta) = \alpha \oplus \beta$ and for $\beta \leq \alpha$, $\alpha \oplus (-\beta) = \alpha \ominus \beta$.

```
1 Precondition: −w_i(a) ≤ α ≤ min_{t∈D^S, t[i]=a} w_S(t);
2 Procedure Project(w_S, i, a, α)
3     w_i(a) ← w_i(a) ⊕ α;
4     foreach (t ∈ D^S such that t[i] = a) do
5         w_S(t) ← w_S(t) ⊖ α;
```

To enforce T-DAC on a cost function $w_S$, it suffices to first shift the cost of every unary cost function $w_i, i \in S$ inside $w_S$ by applying Project$(w_S, i, a, -w_i(a))$ for every value $a \in D_i$. Let $j$ be the maximum variable in $S$ according to $\prec$, one can then apply Project$(w_S, j, b, \alpha)$ for every value $(j, b)$ and $\alpha = \min_{t \in D^S, t[j]=b} w_S(t)$. Let $t$ be a tuple where this minimum is reached. $t$ is then a full support for $(j, b)$: $w_j(b) = w_S(t) \bigoplus_{i \in S} w_i(t[i])$. This support can only be broken if for some unary cost functions $w_i, i \in S, i \neq j$, $w_i(a)$ increases for some value $(i, a)$.

To enforce T-DAC on a complete CFN $(X, W)$, one can simply sort $W$ according to $\prec$ and apply the previous process on each cost function, successively. When a cost function $w_S$ is processed, all the cost functions whose maximum variable appears before the maximum variable of $S$ have already been processed which guarantees that none of the established full supports will be broken in the future. Enforcing T-DAC is therefore in $O(ed^r)$ in time, where $e = |W|$ and $r = \max_{w_S \in W} |S|$ . Using the $\Delta$ data-structures introduced in [7], space can be reduced to $O(edr)$.

The most efficient algorithms for enforcing VAC enforces an approximation of VAC called VAC$_\varepsilon$ with a time complexity in $O(\frac{ekd^r}{\varepsilon})$ and a space complexity in $O(edr)$. Alternatively, optimal soft arc consistency can be used to enforce VAC in $O(e^{6.5} d^{(3r+3.5)} \log M)$ time (where $M$ is the maximum finite cost in the network).

### 3.2 Berge acyclicity and directional arc consistency

In this section, we show that enforcing T-DAC on a Berge-acyclic decomposition of a cost function or on the original global cost function yields the same cost distribution on the last variable and therefore the same lower bound (obtained by node consistency [16]).

**Theorem 2** *If a global cost function $z(T, \theta)$ decomposes into a Berge-acyclic CFN $N = (T \cup E, F)$ then there is an ordering on $T \cup E$ such that the unary cost function $w_{i_n}$ on the last variable $i_n$ produced by enforcing T-DAC on the sub-network $(T, \{z(T, \theta)\} \cup$*

$\{w_i\}_{i\in T}$) *is identical to the unary cost function* $w'_{i_n}$ *produced by enforcing T-DAC on the decomposition* $N = (T \cup E, F \cup \{w_i\}_{i\in T})$.

**Proof** Consider the decomposed network $N$ and $I_N = (T \cup E \cup F, E_I)$ its incidence graph. We know that $I_N$ is a tree whose vertices are the variables and the cost functions of $N$. We root $I_N$ in a variable of $T$. The neighbors (parent and children, if any) of a cost functions $w_S$ are the variables in $S$. The neighbors of a variable $i$ are the cost functions involving $i$. Consider any topological ordering of the vertices of $I_N$. This ordering induces a variable ordering $(i_1, \ldots, i_n), i_n \in T$ which is used to enforce T-DAC on $N$. Notice that for any cost function $w_S \in F$, the parent variable of $w_S$ in $I_N$ appears after all the other variables of $S$.

Consider a value $(i_n, a)$ of the root. If $w_{i_n}(a) = k$, then any complete assignment extending this value has cost $w_{i_n}(a)$. Otherwise, $w_{i_n}(a) < k$. Let $w_S$, be any child of $i_n$ and $t_S$ a full support of $(i_n, a)$ on $w_S$. We have $w_{i_n}(a) = w_S(t) \bigoplus_{i\in S} w_i(t[i])$ which proves that $w_S(t) = 0$ and $\forall i \in S, i \neq i_n, w_i(t[i]) = 0$. $I_N$ being a tree, we can inductively apply the same argument on all the descendants of $i_n$ until leaves are reached, proving that the assignment $(i_n, a)$ can be extended to a complete assignment with cost $w_{i_n}(a)$ in $N$. In either case, $w_{i_n}(a)$ is the cost of an optimal extension of $(i_n, a)$ in $N$.

Suppose now that we enforce T-DAC using the previous variable ordering on the undecomposed sub-network $(T, \{z(T, \theta)\} \cup \{w_i\}_{i\in T})$. Let $t$ be a full support of $(i_n, a)$ on $z(T, \theta)$. By definition $w_{i_n}(a) = z(T, \theta) \bigoplus_{i\in T} w_i(t[i])$ which proves that $w_{i_n}(a)$ is the cost of an optimal extension of $(i_n, a)$ on $(T, \{z(T, \theta)\} \cup \{w_i\}_{i\in T})$. By definition of decomposition, and since $i_n \notin E$, this is equal to the cost of an optimal extension of $(i_n, a)$ in $N$. □

T-DAC has therefore enough power to handle Berge-acyclic decompositions without losing any filtering strength, provided a correct order is used for applying EPTs.

### 3.3 Berge acyclicity and virtual arc consistency

Virtual Arc Consistency offers a simple and direct link between CNs and CFNs which allows to directly lift classical CNs properties to CFNs, under simple conditions.

**Theorem 3** *In a CFN, if a global cost function* $z(T, \theta)$ *decomposes into a Berge-acyclic CFN* $N = (T \cup E, F)$ *then enforcing VAC on either* $(T, \{z(T, \theta)\} \cup \{w_i\}_{i\in T})$ *or on* $(T \cup E, F \cup \{w_i\}_{i\in T})$ *yields the same lower bound* $w_\varnothing$.

**Proof** Enforcing VAC on the CFN $P = (T \cup E, F \cup \{w_i\}_{i\in T})$ does not modify the set of scopes and yields an equivalent problem $P'$ such that $Bool(P')$ is Berge-acyclic, a situation where arc consistency is a decision procedure. We can directly make use of Proposition 10.5 of [7] which states that if a CFN $P$ is VAC and if $Bool(P)$ is in a class of CSPs for which arc consistency is a decision procedure, then $P$ has an optimal solution of cost $w_\varnothing$.

Similarly, the network $Q = (T, \{z(T, \theta)\} \cup \{w_i\}_{i\in T})$ contains just one cost function with arity strictly above 1 and $Bool(Q)$ will be decided by arc consistency. Enforcing VAC will therefore provide a CFN which also has an optimal solution of cost $w_\varnothing$. The networks $P$ and $Q$ having the same optimal cost by definition of a decomposition. □

## 4 Experimental Results

In this section, we intend to evaluate the practical interest of global cost function decompositions. Compared to the monolithic cost function filtering algorithm, these decompositions allow for a simple implementation and will provide effective filtering. But their actual performance needs to be evaluated.

All problems were solved using the CFN solver `toulbar2 0.9.5`[3] with pre-processing off (option line `-o -e: -f: -dec: -h: -c: -d: -q:`), and a variable assignment and DAC ordering compatible with the Berge-acyclic structure of the decompositions. The dynamic value ordering chooses the existential EAC value first [15]. No initial upper bound is used. The same level of local consistency (namely (weak) EDGAC*, stronger than T-DAC and which therefore will produce an optimal $w_\varnothing$ for every global cost function) was used in all cases. All the experiments were run using several 2.66 Ghz Intel Xeon CPU cores with 64GB RAM.

### 4.1 Random WEIGHTEDREGULAR

Following [21], we generated random automata with $|Q|$ states and $|\Sigma|$ symbols. We randomly selected $30\%$ of all possible pairs $(s, q_i) \in \Sigma \times Q$ and randomly chose a state $q_j \in Q$ to form a transition $\delta(s, q_i) = q_j$ for each such pair. The set of final states $F$ is obtained by randomly selecting $50\%$ of states in $Q$. Random sampling uses a uniform distribution.

From each automaton, we built two CFNs: one using a monolithic SOFTREGULAR cost function using Hamming distance [19] and another using the Berge-acyclic decomposition of an equivalent WEIGHTEDREGULAR global cost functions. To make the situation more realistic, we added to each of these problems the same set of random unary constraints (one per non-auxiliary variable, unary costs randomly chosen between 0 and 9). We measured two times: (1) time for loading and filtering the initial problem and (2) total time for solving the CFN (including the previous time). The first time is informative on the filtering complexity while the second emphasizes the incrementality of the filtering algorithms. Times were averaged on 100 runs and samples reaching the time limit of one hour were counted as such.

| $n$ | $\|\Sigma\|$ | $\|Q\|$ | Monolithic | | Decomposed | |
|---|---|---|---|---|---|---|
| | | | filter | solve | filter | solve |
| 25 | 5 | 10 | 0.12 | 0.51 | 0.00 | 0.00 |
| | | 80 | 2.03 | 9.10 | 0.08 | 0.08 |
| 25 | 10 | 10 | 0.64 | 2.56 | 0.01 | 0.01 |
| | | 80 | 10.64 | 43.52 | 0.54 | 0.56 |
| 25 | 20 | 10 | 3.60 | 13.06 | 0.03 | 0.03 |
| | | 80 | 45.94 | 177.5 | 1.51 | 1.55 |
| 50 | 5 | 10 | 0.45 | 3.54 | 0.00 | 0.00 |
| | | 80 | 11.85 | 101.2 | 0.17 | 0.17 |
| 50 | 10 | 10 | 3.22 | 20.97 | 0.02 | 0.02 |
| | | 80 | 51.07 | 380.5 | 1.27 | 1.31 |
| 50 | 20 | 10 | 15.91 | 100.7 | 0.06 | 0.07 |
| | | 80 | 186.2 | 1,339 | 3.38 | 3.47 |

Looking just to filtering time, it is clear that decomposition offers impressive improvements despite a much simpler implementation. Solving times show that it also inherits the excellent incrementality of usual consistency enforcing algorithms for free.

[3] https://mulcyber.toulouse.inra.fr/projects/toulbar2.

## 4.2 Nonograms

(prob012 in the CSPLib) are NP-complete logic puzzles in which cells in a grid have to be colored in such a way that a given description for each row and column, giving the lengths of distinct colored segments, is adhered to.

A $n \times n$ nonogram can be represented using $n^2$ Boolean variables $x_{ij}$ specifying the color of the square at position $(i, j)$. The restrictions on the lengths of segments in each row or column can be captured by a REGULAR constraint. In order to evaluate the interest of filtering decomposable cost functions, we have performed two types of experiments on nonograms.

**Softened nonograms:** can be built from classical nonograms by relaxing the strict adherence to the indicated lengths of colored segments. For this, we relax the REGULAR constraints on each row and column in the softened version using the Hamming distance. The associated cost indicates how many cells need to be modified to satisfy the attached description. This problem contains $2n$ WEIGHTEDREGULAR cost functions, with intersecting scopes. In order to be able to apply Theorem 2 on each of these global cost functions, one must build a global variable order which is a topological ordering for each of these cost functions. Although this requirement seems hard to meet in general, it is easy to produce in this specific case. The $x_{ij}$ variables can, for example, be ordered in lexicographic order, from top left to bottom right and auxiliary variables inserted anywhere between their flanking original variables. Global cost function scopes are usually expressed to capture properties defined on time (as in rostering problems) or space (as in nonograms, or text processing problems). In those cases, the global order defined by time or space defines a global variable ordering that will often satisfy the conditions of Theorem 2.

Random $n \times n$ nonogram instances are generated by uniformly sampling the number of segments in each row/column between 1 and $\lfloor \frac{n}{3} \rfloor$. The length of each segment is uniformly and iteratively sampled from 1 to the maximum length that allows remaining segments to be placed (considering a minimum length of 1).

We solved these problems with `toulbar2` as before and measured the percentage of problems solved as well as the mean cpu-time (unsolved problems are counted for one hour) on samples of 100 problems.

| Size | Monolithic | | Decomposed | |
|---|---|---|---|---|
| | Solved | Time | Solved | Time |
| $6 \times 6$ | 100% | 1.98 | 100% | 0.00 |
| $8 \times 8$ | 96% | 358 | 100% | 0.52 |
| $10 \times 10$ | 44% | 2,941 | 100% | 30.2 |
| $12 \times 12$ | 2% | 3,556 | 82% | 1,228 |
| $14 \times 14$ | 0% | 3,600 | 14% | 3,316 |

In this more realistic setting, involving different interacting global cost functions, decomposition is again the most efficient approach with orders of magnitude speedups.

**White noise images:** a random solution grid, with each cell colored with probability 0.5, is generated. A nonogram problem instance is created from the lengths of the segments observed in this random grid. These problems usually have several solutions, among which the original grid. We associate random unary costs, uniformly sampled betwen 0 and 99, with each cell. These costs represent the price to color the cell. A solution with minimum cost is sought. This problem has been modeled in `choco` (rel. 2.1.3, default options) and `toulbar2` (`-h:` option) using $2n$ REGULAR global constraints. In the `choco` model, a SCALAR constraint involving all variables is used to define the criteria to optimize. In `toulbar2`, coloring costs are captured by unary cost functions and the REGULAR constraints are represented by WEIGHTEDREGULAR cost functions with weights in $\{0, k\}$. The monolithic version has been tried but gave very poor results.

We measured the percentage of problems solved as well as the mean cpu-time (unsolved problems are counted for $\frac{1}{2}$ hour, the time-limit used) on samples of 50 problems.

| Size | `choco` | | `toulbar2` | |
|---|---|---|---|---|
| | Solved | Time | Solved | Time |
| $20 \times 20$ | 100% | 1.88 | 100% | 0.93 |
| $25 \times 25$ | 100% | 14.78 | 100% | 3.84 |
| $30 \times 30$ | 96% | 143.6 | 96% | 99.01 |
| $35 \times 35$ | 80% | 459.9 | 94% | 218.2 |
| $40 \times 40$ | 46% | 1,148 | 66% | 760.8 |
| $45 \times 45$ | 14% | 1,627 | 32% | 1.321 |

On this problem, enforcing soft filtering on decomposed global cost functions is preferable to traditional bound/GAC filtering of a pure CP model with cost variables. Using decomposition, the direct use of soft filtering such as EDAC, which subsumes T-DAC, provides a better exploitation of costs, with minimal implementation efforts.

## Beyond decomposable cost functions

In some cases, problems may contain global cost functions which are not decomposable just because the bounded arity cost function decomposition is not polynomial in size. However, if the network is Berge-acyclic, Theorem 2 still applies. With exponential size networks, filtering will take exponential time but may yield strong lower bounds. The linear equation global constraint $\sum_{i=1}^{n} a_i x_i = b$ ($a$ and $b$ being small integer coefficients) can be easily decomposed introducing $n-3$ intermediate sum variables $q_i$ and ternary sum constraints of the form $q_{i-1} + a_i x_i = q_i$ with $i \in [3, n-2]$ and $a_1 x_1 + a_2 x_2 = q_2$, $q_{n-2} + a_{n-1} x_{n-1} + a_n x_n = b$. The auxiliary variables $q_i$ have $b$ values which is exponential in the representation of $b$. We consider the Market Split problem defined in [10, 25]. The goal is to minimize $\sum_{i=1}^{n} o_i x_i$ such that $\sum_{i=1}^{n} a_{i,j} x_i = b_j$ for each $j \in [1, m]$ and $x_i$ are Boolean variables in $\{0, 1\}$ ($o$, $a$ and $b$ being positive integer coefficients). We compared the Berge-acyclic decomposition in `toulbar2` with a direct application of the Integer Linear Programming solver `cplex` (version 12.2.0.0). We generated random instances with random integer coefficients in $[0, 99]$ for $o$ and $a$, and $b_j = \lfloor \frac{1}{2} \sum_{i=1}^{n} a_{i,j} \rfloor$. We used a sample of 50 problems with $m = 4, n = 30$ leading to $\max b_j = 918$. The mean number of nodes developed in `cplex` is 50% higher than in `toulbar2`. But `cplex` was on average 6 times faster than `toulbar2` on these problems. 0/1 knapsack problems probably represent a worst case situation for `toulbar2` given that `cplex` embeds much of what is known about 0/1 knapsacks (and only part of these extend to more complicated domains). Possible avenues to improve `toulbar2` results in this unfavorable situation would be to use a combination of the $m$ knapsack constraints into one as suggested in [25] and a direct exploitation of the properties of the ternary linear constraints for more compact representation and more efficient filtering.

## Related works

It should be pointed out that T-DAC is closely related to mini-buckets [12] and Theorem 2 can easily be adapted to this scheme.

Mini-buckets perform a weakened form of variable elimination: when a variable $x$ is eliminated, the cost functions linking $x$ to the remaining variables are partitioned into sets containing at most $i$ variables in their scopes and at most $m$ functions. If we compute mini-buckets using the same variable ordering, with $m = 1$ and unbounded $i$, we will obtain the same marginal cost function as T-DAC on the root variable $r$, with the same time complexity. Mini-buckets can be used along two main recipes: precomputed (static) mini-buckets do not require update during search but restrict search to one static variable ordering; dynamic mini-buckets allow for dynamic variable ordering (DVO) but suffer from a lack of incrementality. Soft local consistencies, being based on EPTs, always yield equivalent problems, providing incrementality during search and are compatible with DVO. Soft arc consistencies also offer a space complexity in $O(edr)$ while mini-bucket may require space exponential in $i$.

## Conclusion

In this paper, we have extended constraint decomposition to cost functions occurring in CFNs. For cost functions having a Berge-acyclic decomposition, we have shown that a simple filtering, at the directed arc consistency level, provides a comparable filtering on the decomposition or on the global cost function itself, provided a suitable variable ordering is used for DAC enforcing. For the stronger Virtual AC filtering, the same result is obtained, without any requirement.

The application of this result on the trivial class of Berge-acyclic global cost functions defined by Berge-acyclic decomposable global *constraints* is already significant since it allows to enforce *soft* local consistencies on networks containing Berge-acyclic decomposable global constraints such as REGULAR, GRAMMAR, AMONG,...

We have shown that these Berge-acyclic global constraints can also be relaxed into a Berge-acyclic global cost function using a generalization of the usual "decomposition" measure. This immediately provides a long list of Berge-acyclic decomposable global *cost functions*. Our experimental results based on the application of DAC on the relaxation of the REGULAR constraint into the WEIGHTEDREGULAR cost function show that the decomposition approach offers impressive speedups and cheap implementation compared to the monolithic cost function algorithms.

To experimentally evaluate the practical interest of the stronger result on VAC, a technically involved implementation of VAC on non binary constraints would be needed.

Although it is currently restricted to Berge-acyclic decompositions, this work paves the way for a more general form of "structural decompositions" of global cost functions where global cost functions decompose into an acyclic structure of local cost functions, with bounded separator sizes (but not necessarily of cardinality 1). These global structurally decomposed cost functions could then be filtered efficiently through dedicated incremental equivalence preserving transformations capturing non serial dynamic programming algorithms.

## REFERENCES


[1] C. Beeri, R. Fagin, D. Maier, and M.Yannakakis, 'On the desirability of acyclic database schemes', *Journal of the ACM*, **30**, 479–513, (1983).

[2] N. Beldiceanu, M. Carlsson, R. Debruyne, and T. Petit, 'Reformulation of global constraints based on constraints checkers', *Constraints*, **10**(4), 339–362, (2005).

[3] C. Bessiere, 'Constraint propagation', in *Handbook of Constraint Programming*, eds., F. Rossi, P. van Beek, and T. Walsh, chapter 3, Elsevier, (2006).

[4] C. Bessiere, E. Hebrard, B. Hnich, Z. Kiziltan, and T. Walsh, 'Slide: A useful special case of the cardpath constraint', in *Proc. of ECAI'08*, pp. 475–479, (2008).

[5] C. Bessiere and P. Van Hentenryck, 'To be or not to be ... a global constraint', in *Proc. CP'03*, pp. 789–794, (2003).

[6] E. Boros and P. Hammer, 'Pseudo-Boolean Optimization', *Discrete Appl. Math.*, **123**, 155–225, (2002).

[7] M. Cooper, S. de Givry, M. Sanchez, T. Schiex, M. Zytnicki, and T. Werner, 'Soft arc consistency revisited', *Artificial Intelligence*, **174**, 449–478, (2010).

[8] M C. Cooper, 'Reduction operations in fuzzy or valued constraint satisfaction', *Fuzzy Sets and Systems*, **134**(3), 311–342, (2003).

[9] M C. Cooper and T. Schiex, 'Arc consistency for soft constraints', *Artificial Intelligence*, **154**(1-2), 199–227, (2004).

[10] Gérard Cornuéjols and Milind Dawande, 'A class of hard small 0-1 programs', in *Integer Programming and Combinatorial Optimization, 6th International IPCO Conference, Houston, Texas, USA, June 22-24, 1998, Proceedings*, pp. 284–293, (1998).

[11] Karel Culik II and Jarkko Kari, 'Image compression using weighted finite automata', in *MFCS*, eds., Andrzej M. Borzyszkowski and Stefan Sokolowski, volume 711 of *Lecture Notes in Computer Science*, pp. 392–402. Springer, (1993).

[12] Rina Dechter, 'Mini-buckets: A general scheme for generating approximations in automated reasoning', in *Proc. of the* $16^{th}$ *IJCAI*, pp. 1297–1303, (1997).

[13] George Katsirelos, Nina Narodytska, and Toby Walsh, 'The weighted grammar constraint', *Annals OR*, **184**(1), 179–207, (2011).

[14] D. Koller and N. Friedman, *Probabilistic graphical models*, MIT press, 2009.

[15] J. Larrosa, S. de Givry, F. Heras, and M. Zytnicki, 'Existential arc consistency: getting closer to full arc consistency in weighted CSPs', in *Proc. of the* $19^{th}$ *IJCAI*, pp. 84–89, Edinburgh, Scotland, (August 2005).

[16] J. Larrosa and T. Schiex, 'In the quest of the best form of local consistency for weighted CSP', in *Proc. of the* $18^{th}$ *IJCAI*, pp. 239–244, Acapulco, Mexico, (August 2003).

[17] Javier Larrosa and Thomas Schiex, 'Solving weighted CSP by maintaining arc consistency', *Artif. Intell.*, **159**(1-2), 1–26, (2004).

[18] JHM. Lee and KL. Leung, 'Consistency techniques for flow-based projection-safe global cost functions in weighted constraint satisfaction', *Journal of Artificial Intelligence Research*, **43**, 257–292, (2012).

[19] Jimmy Ho-Man Lee and Ka Lun Leung, 'Towards efficient consistency enforcement for global constraints in weighted constraint satisfaction', in *Proc of the* $21^{th}$ *IJCAI*, ed., Craig Boutilier, pp. 559–565, (2009).

[20] C.H. Papadimitriou and M. Yannakakis, 'Optimization, approximation, and complexity classes', *Journal of Computer and System Sciences*, **43**(3), 425–440, (1991).

[21] Gilles Pesant, 'A regular language membership constraint for finite sequences of variables', in *CP*, ed., Mark Wallace, volume 3258 of *Lecture Notes in Computer Science*, pp. 482–495. Springer, (2004).

[22] Thierry Petit, Jean-Charles Régin, and Christian Bessiere, 'Specific filtering algorithms for over-constrained problems', in *CP*, pp. 451–463, (2001).

[23] Martí Sánchez, Simon de Givry, and Thomas Schiex, 'Mendelian error detection in complex pedigrees using weighted constraint satisfaction techniques', *Constraints*, **13**(1-2), 130–154, (2008).

[24] T. Schiex, H. Fargier, and G. Verfaillie, 'Valued constraint satisfaction problems: hard and easy problems', in *Proc. of the* $14^{th}$ *IJCAI*, pp. 631–637, Montréal, Canada, (August 1995).

[25] M. A. Trick, 'A dynamic programming approach for consistency and propagation for knapsack constraints', *Annals of Operations Research*, **118**(1-4), 73–84, (2003).

[26] Willem Jan van Hoeve, Gilles Pesant, and Louis-Martin Rousseau, 'On global warming: Flow-based soft global constraints', *J. Heuristics*, **12**(4-5), 347–373, (2006).

[27] M. Zytnicki, C. Gaspin, S. de Givry, and T. Schiex, 'Bounds arc consistency for weighted CSPs', *Journal of Artificial Intelligence Research*, **35**(2), 593–621, (2009).

# Improved Bounded Max-Sum for Distributed Constraint Optimization

**Emma Rollon** and **Javier Larrosa** [1]

**Abstract.** Bounded Max-Sum is a message-passing algorithm for solving Distributed Constraint Optimization Problems able to compute solutions with a guaranteed approximation ratio. Although its approximate solutions were empirically proved to be within a small percentage of the optimal solution on low and moderately dense problems, in this paper we show that its theoretical approximation ratio is overestimated, thus overshadowing its good performance. We propose a new algorithm, called Improved Bounded Max-Sum, whose approximate solutions are at least as good as the ones found by Bounded Max-Sum and with a tighter approximation ratio. Our empirical evaluation shows that the new approximation ratio is significantly tighter.

## 1 Introduction

*Decentralised coordination* techniques are a very important topic of research. A common approach is to cast the problem as a *multi-agent distributed constraint optimization problem* (DCOP), where the possible actions that agents can take are associated with *variables* and the utility for taking joint actions are encoded with (soft) *constraints* [10]. The set of constraints define a global utility function $F(x)$ to be optimized via decentralised coordination of the agents. In general, complete algorithms [7, 6, 9] (i.e. algorithms that find the true optimum) exhibit an exponentially increasing coordination overhead, which makes them useless in many practical situations.

Approximate algorithms constitute a very interesting alternative. They require little computation and communication at the cost of sacrificing optimality. There are several examples showing that they can provide solutions which are very close to optimality [3, 5]. However, this observation can only be verified on small toy instances, because it requires the computation of the true optimal to compare with, and it is not available in real-size real-world situations.

A significant breakthrough along this line of work was the Bounded Max-Sum algorithm (BMS) [10]. This algorithm comes with a guarantee approximation ratio $\widetilde{\rho}$, meaning that its approximate solution $\widetilde{\mathbf{x}}$ has a utility $F(\widetilde{\mathbf{x}})$ which is no more than a factor $\widetilde{\rho} \geq 1$ away from the optimum (i.e, $F(\widetilde{\mathbf{x}}) \leq F(\mathbf{x}^*) \leq \widetilde{\rho}F(\widetilde{\mathbf{x}})$). Clearly, large values of $\widetilde{\rho}$ reflect lack of confidence in the solution $\widetilde{\mathbf{x}}$. There are two possible reasons for a large $\widetilde{\rho}$: $i$) the algorithm failed in finding a solution close to the optimal, $ii$) the approximation ratio is not tight. Clearly, if we want $\widetilde{\rho}$ to be our measure of confidence about the quality of $\widetilde{\mathbf{x}}$, we want a tight $\widetilde{\rho}$ (i.e, $F(\mathbf{x}^*) \approx \widetilde{\rho}F(\widetilde{\mathbf{x}})$). Thus, the quality of the approximation ratio is a matter of the utmost importance.

[1] Departament de Llenguatges i Sistemes Informàtics, Universitat Politècnica de Catalunya, Spain.

In this paper we propose an improvement of BMS with approximation ratio $\rho$. We theoretically show that it is always better than the previous one (i.e., $\rho \leq \widetilde{\rho}$). Moreover, our experiments show that, in practice, $\rho$ is much tighter than $\widetilde{\rho}$.

## 2 Preliminaries

In this Section we review the main elements to contextualize our work. Definitions and notation is borrowed almost directly from [10]. We urge the reader to visit that reference for more details and examples.

### 2.1 DCOP

A *Distributed Constraint Optimization Problem* (DCOP) is a quadruple $P = (\mathbf{A}, \mathbf{X}, \mathbf{D}, \mathbf{F})$, where $\mathbf{A} = \{\mathcal{A}_1, \dots, \mathcal{A}_r\}$ is a set of agents, and $\mathbf{X} = \{x_1, \dots, x_n\}$ and $\mathbf{D} = \{\mathbf{d}_1, \dots, \mathbf{d}_n\}$ are variables and domains. $\mathbf{F} = \{f_1, \dots, f_e\}$ is a set of cost functions. The objective function is,

$$F(x) = \sum_{j=1}^{e} f_j(x^j)$$

where $x^j \subseteq \mathbf{X}$ is the scope of $f_j$. A *solution* is a complete assignment $\mathbf{x}$. An *optimal solution* is a complete assignment $\mathbf{x}^*$ such that $\forall \mathbf{x},\ F(\mathbf{x}^*) \geq F(\mathbf{x})$. The usual task of interest is to find $\mathbf{x}^*$ through the coordination of the agents.

In the applications under consideration, the agents search for the optimum via decentralised coordination. We assume that each agent can control only its local variable(s) and has knowledge of, and can directly communicate with, a few neighboring agents. Two agents are neighbors if there is a relationship connecting variables and functions that the agents control.

The structure of a DCOP problem $P = (\mathbf{A}, \mathbf{X}, \mathbf{D}, \mathbf{F})$ is represented by its associated factor graph. A *factor graph* is a bipartite graph having a variable node for each variable $x_i \in \mathbf{X}$, a factor node for each local function $f_j \in \mathbf{F}$, and an edge connecting variable node $x_i$ to factor node $f_j$ if and only if $x_i$ is an argument of $f_j$.

### 2.2 Max-Sum Algorithm

The *Max-Sum* algorithm [2, 1] is a message-passing algorithm for solving DCOP problems. It operates over a factor graph by sending functions (a.k.a., messages) along its edges. Edge $(i, j)$ has associated two messages $q_{i\to j}$, from variable node $x_i$ to function node $f_j$, and $r_{j\to i}$, from function node $f_j$ to variable node $x_i$. These messages are defined as follows:

- **From variable to function:**

$$q_{i\to j}(x_i) = \alpha_{ij} + \sum_{k\in\mathcal{M}_i\setminus j} r_{k\to i}(x_i)$$

where $\mathcal{M}_i$ is a vector of function indexes, indicating which function nodes are connected to variable node $x_i$, and $\alpha_{ij}$ is a normalizing constant to prevent the messages from increasing endlessly in cyclic graphs.

- **From function to variable:**

$$r_{j\to i}(x_i) = \max_{x^j\setminus x_i}\{f_j(x^j) + \sum_{k\in\mathcal{N}_j\setminus i} q_{k\to i}(x_i)\}$$

where $\mathcal{N}_j$ is a vector of variable indexes, indicating which variable nodes are connected to function node $f_j$ and $x^j \setminus x_i = \{x_k \mid k \in \mathcal{N}_j \setminus i\}$

Max-Sum is a distributed synchronous algorithm, since the agent controlling node $i$ has to wait to receive messages from all its neighbors but $j$, to be able to compute (and send) its message to $j$. When the factor graph is cycle free, the algorithm is guaranteed to converge to the global optimal solution. Once the convergence is reached, each variable node can compute function,

$$z_i(x_i) = \max_{x_i} \sum_{k\in\mathcal{M}_i} r_{k\to i}(x_i)$$

The optimal solution is $\max_{x_i}\{z_i(x_i)\}$ and the optimal assignment $\mathbf{x}_i^* = \arg\max_{x_i}\{z_i(x_i)\}$. When the factor graph is cyclic, the algorithm may not converge to the optimum and only provides an approximation.

## 3 Bounded Max-Sum Algorithm

The *Bounded Max-Sum* algorithm (BMS) [10], is an approximation algorithm built on the Max-Sum algorithm. From a possibly cyclic problem $P$, the idea is to remove cycles in its factor graph by ignoring dependencies between functions and variables which have the least impact on the solution quality, producing a new acyclic problem $\widetilde{P}$. Then, Max-Sum is used to optimally solve $\widetilde{P}$ while simultaneously computing the approximation ratio $\widetilde{\rho}$. A more detailed description follows. For the sake of simplicity, we will restrict ourselves to the case of binary functions $f_j(x_i, x_k)$. The extension to general functions is direct. The algorithm works in three phases, each one implementable in a decentralised manner (see [10] for further details):

- **Relaxation Phase:** First, the algorithm weights each edge $(i, j)$ of the original factor graph as,

$$w_{ij} = \max_{x_k}\{\max_{x_i} f_j(x_i, x_k) - \min_{x_i} f_j(x_i, x_k)\}$$

Then, it finds a maximum spanning tree $T$. Let $W$ be the sum of weights of the removed edges (i.e, $W = \sum_{(i,j)\notin T} w_{ij}$). Next, the original problem $P$ is transformed into an acyclic one $\widetilde{P}$ having the spanning tree $T$ as factor graph. This is done as follows: for each edge $(i, j)$ in the original graph that does not belong to the tree, the cost function $f_j(x_i, x_k)$ is transformed into another function $\widetilde{f}_j(x_k)$ defined as,

$$\widetilde{f}_j(x_k) = \min_{x_i} f_j(x_i, x_k)$$

Note that the objective function of $\widetilde{P}$ is

$$\widetilde{F}(x) = \sum_{(i,j),(k,j)\in T} f_j(x_i, x_k) + \sum_{(i,j)\notin T} \widetilde{f}_j(x_k)$$

- **Solving Phase:** BMS solves $\widetilde{P}$ with Max-Sum. Let $\widetilde{\mathbf{x}}$ be the solution of this problem. Since the factor graph of $\widetilde{P}$ is acyclic, $\widetilde{\mathbf{x}}$ is its optimal assignment.
- **Bounding Phase:** In [10], it is proved that,

$$F(\widetilde{\mathbf{x}}) \le F(\mathbf{x}^*) \le \widetilde{F}(\widetilde{\mathbf{x}}) + W$$

We can rewrite the previous upper bound expression as,

$$F(\mathbf{x}^*) \le \frac{\widetilde{F}(\widetilde{\mathbf{x}}) + W}{F(\widetilde{\mathbf{x}})} F(\widetilde{\mathbf{x}})$$

Therefore, the algorithm computes $\widetilde{\rho} = \frac{\widetilde{F}(\widetilde{\mathbf{x}})+W}{F(\widetilde{\mathbf{x}})}$, which is a guarantee approximation ratio.

## 4 Improved BMS

### 4.1 Theoretical elements

Consider an edge $(i, j)$ in the original factor graph that does not belong to the spanning tree. We define $\widehat{f}_j(x_k)$ as,

$$\widehat{f}_j(x_k) = \max_{x_i} f_j(x_i, x_k)$$

Let $\widehat{P}$ denote the problem containing the not modified functions $f_j(x_i, x_k)$ (for $(i, j), (k, j) \in T$) and the $\widehat{f}_j(x_k)$ functions (for $(i, j) \notin T$). Note that $\widehat{P}$ and $\widetilde{P}$ have the same acyclic factor graph. Note as well that the objective function of $\widehat{P}$ is

$$\widehat{F}(x) = \sum_{(i,j),(k,j)\in T} f_j(x_i, x_k) + \sum_{(i,j)\notin T} \widehat{f}_j(x_k)$$

We can solve $\widehat{P}$ with Max-Sum. Let $\widehat{\mathbf{x}}$ be the optimal solution of this problem. It is obvious that $F(\widehat{\mathbf{x}})$ is a lower bound of $F(\mathbf{x}^*)$. Furthermore, as we prove next, $\widehat{F}(\widehat{\mathbf{x}})$ is an upper bound of $F(\mathbf{x}^*)$. Therefore, $\widehat{\rho} = \frac{\widehat{F}(\widehat{\mathbf{x}})}{F(\widehat{\mathbf{x}})}$ is a guarantee approximation ratio.

**Theorem 1** $F(\mathbf{x}^*) \le \widehat{F}(\widehat{\mathbf{x}})$.

**Proof** By definition, $F(\mathbf{x}^*) = \sum_{(i,j),(k,j)\in T} f_j(\mathbf{x}_i^*, \mathbf{x}_k^*) + \sum_{(i,j)\notin T} f_j(\mathbf{x}_i^*, \mathbf{x}_k^*)$. Since for all $f_j$ we have that $f_j(x_i, x_k) \le \max_{x_i} f_j(x_i, x_k)$, then

$$F(\mathbf{x}^*) \le \sum_{(i,j),(k,j)\in T} f_j(\mathbf{x}_i^*, \mathbf{x}_k^*) + \sum_{(i,j)\notin T} \max_{x_i} f_j(x_i, \mathbf{x}_k^*) = \widehat{F}(\mathbf{x}^*)$$

From the optimality of $\widehat{\mathbf{x}}$, we know that $\widehat{F}(\mathbf{x}^*) \le \widehat{F}(\widehat{\mathbf{x}})$, which proves the theorem.

Next, we show that $\widehat{F}(\widehat{\mathbf{x}})$ is a tighter upper bound than $\widetilde{F}(\widetilde{\mathbf{x}}) + W$.

**Theorem 2** $\widehat{F}(\widehat{\mathbf{x}}) \le \widetilde{F}(\widetilde{\mathbf{x}}) + W$.

**Proof** The proof is direct once it has been noted that for all $f_j(x_i, x_k)$,

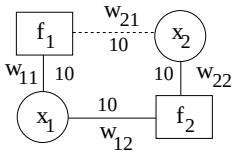
$$\widehat{f}_j(x_k) \le \widetilde{f}_j(x_k) + w_{ij}$$

which we prove next. By definition, the previous equation corresponds to,

$$\max_{x_i} f_j(x_i, x_k) \le \min_{x_i} f_j(x_i, x_k) + \max_{x_k}\{\max_{x_i} f_j(x_i, x_k) - \min_{x_i} f_j(x_i, x_k)\}$$

which can be rewritten as,

$$\max_{x_i} f_j(x_i, x_k) - \min_{x_i} f_j(x_i, x_k) \le \max_{x_k}\{\max_{x_i} f_j(x_i, x_k) - \min_{x_i} f_j(x_i, x_k)\}$$




| $x_1$ | $x_2$ | $f_1$ |
|---|---|---|
| a | a | 10 |
| a | b | 10 |
| b | a | 0 |
| b | b | 10 |

| $x_1$ | $x_2$ | $f_2$ |
|---|---|---|
| a | a | 10 |
| a | b | 0 |
| b | a | 0 |
| b | b | 10 |

| $x_1$ | $\widetilde{f}_1$ |
|---|---|
| a | 10 |
| b | 0 |

| $x_1$ | $\widehat{f}_1$ |
|---|---|
| a | 10 |
| b | 10 |

**Figure 1.** Example of a factor graph containing cycles and a spanning tree formed by removing the edge between variable node $x_2$ and function node $f_1$.

which clearly holds.

We cannot establish any dominance relation between $\widetilde{\rho}$ and $\widehat{\rho}$ because there is no dominance between $F(\widetilde{\mathbf{x}})$ and $F(\widehat{\mathbf{x}})$. However, one way to circumvent this situation is to take $\rho = \frac{\widehat{F}(\widehat{\mathbf{x}})}{\max\{F(\widetilde{\mathbf{x}}),F(\widehat{\mathbf{x}})\}}$. The new ratio $\rho$ dominates $\widetilde{\rho}$.

**Theorem 3** $\rho \leq \widetilde{\rho}$.

**Proof** Direct from Theorem 2 and the fact that $\max\{F(\widetilde{\mathbf{x}}), F(\widehat{\mathbf{x}})\} \geq F(\widetilde{\mathbf{x}})$.

## 4.2 IBMS

*Improved BMS* (IMBS) works, as BMS, in three phases:

- **Relaxation Phase:** IBMS computes the spanning tree $T$ and the relaxed problem $\widetilde{P}$ exactly as BMS does. Additionally, IBMS computes the relaxed problem $\widehat{P}$.
- **Solving Phase:** IBMS solves $\widetilde{P}$ and $\widehat{P}$ with Max-Sum. Let $\widetilde{\mathbf{x}}$ and $\widehat{\mathbf{x}}$ be the solutions of these problems. The agents will act according to the best solution ($\max\{F(\widetilde{\mathbf{x}}), F(\widehat{\mathbf{x}})\}$).
- **Bounding Phase:** IBMS computes the approximation ratio $\rho = \frac{\widehat{F}(\widehat{\mathbf{x}})}{\max\{F(\widetilde{\mathbf{x}}),F(\widehat{\mathbf{x}})\}}$.

The computation, storage and communication effort of IBMS is essentially twice that of BMS, because it requires solving two relaxed problems with Max-Sum. Given the low cost of BMS, doubling it seems acceptable. However, when it is not the case, one can always run a weaker version of IBMS ignoring $\widetilde{P}$. This weaker version will be exactly as costly as BMS. Its disadvantage is that $\widehat{\mathbf{x}}$ is not guaranteed to be better than $\widetilde{\mathbf{x}}$. In fact, our experiments show that there is no clear winner among them. Interestingly, the approximation ratio of the weaker version $\widehat{\rho}$ is systematically better than the approximation ratio of BMS $\widetilde{\rho}$.

**Example 1** *Consider the problem $P$ given in Figure 1 with two variables $\{x_1, x_2\}$ and two functions $\{f_1, f_2\}$. The spanning tree of its factor graph is given with solid lines (i.e., edge $(x_2, f_1)$ has been removed, shown as a dashed line). Thus, $W = 10$. Functions $\widetilde{f}_1$ and $\widehat{f}_1$ in $\widetilde{P}$ and $\widehat{P}$, respectively, are given in the figure. Max-Sum finds assignments $\widetilde{\mathbf{x}} = \widehat{\mathbf{x}} = (x_1 = a, x_2 = a)$, with utility $\widetilde{F}(\widetilde{\mathbf{x}}) = \widehat{F}(\widehat{\mathbf{x}}) = 20$. Their evaluation on the original problem $P$ is $F(\widetilde{\mathbf{x}}) = F(\widehat{\mathbf{x}}) = 20$. The approximation ratios are $\widetilde{\rho} = 1.5$, $\widehat{\rho} = 1$, and $\rho = 1$.*

## 5 Empirical Evaluation

The purpose of the experiments is to evaluate the improvement of our upper bound $\widehat{F}(\widehat{\mathbf{x}})$ and approximation ratios $\rho$ and $\widehat{\rho}$ over the BMS upper bound $\widetilde{F}(\widetilde{\mathbf{x}}) + W$ and approximation ratio $\widetilde{\rho}$, respectively. We

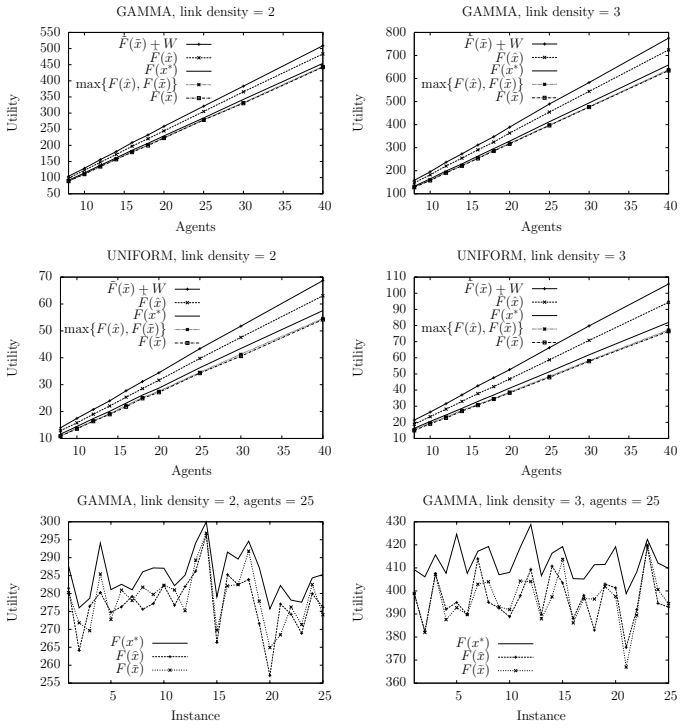



**Figure 2.** First and second row, bounds obtained by algorithms IBMS and BMS varying the number of agents; third row, lower bound detail for instances with 25 agents and gamma distribution.

consider the same set of problems from the ADOPT repository[2] used in [10]. These problems represent graph colouring problems with two different link densities (i.e., the average connection per agent) and different number of nodes. Each agent controls one node (i.e., variable), with domain $|\mathbf{d}_i| = 3$, and each edge of the graph represents a pairwise constraint between two agents. Each edge is associated with a random payoff matrix, specifying the payoff that both agents will obtain for every possible combination of their variables' assignments. Each entry of the payoff matrix is a real number sampled from two different distributions: a gamma distribution with $\alpha = 9$ and $\beta = 2$, and a uniform distribution with range $(0, 1)$. For each configuration, we report average values over 25 repetitions. For the sake of comparison, we compute the optimal utility by a complete centralized algorithm, although this value can only be computed up to 12 agents by a complete decentralized algorithm, as shown in [10].

Figure 2 (first and second rows) shows the upper and lower bound

[2] http://teamcore.usc.edu/dcop

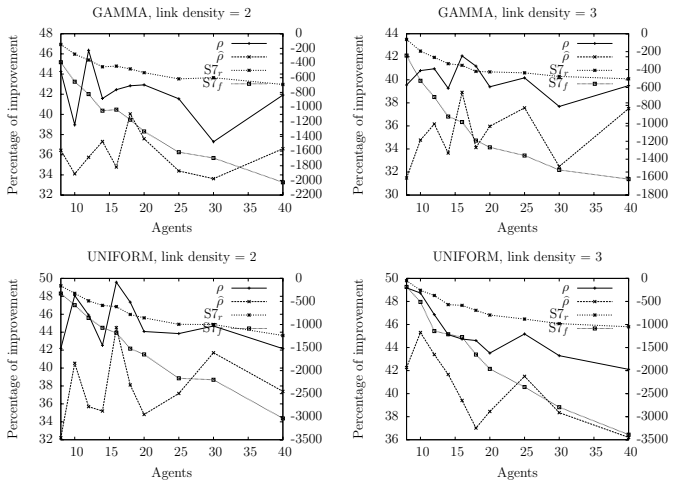



**Figure 3.** Percentage of improvement of the approximation ratio of IBMS $\rho$ and weaker version of IBMS $\widehat{\rho}$ (left y-axe), and percentage of decrease of the 7-size-bounded-distance criteria using the the minimum maximum reward bound ($S7_r$) and the minimum fraction bound ($S7_f$) (right y-axe) over the approximation ratio of BMS $\widetilde{\rho}$.

obtained by IBMS (i.e., $\widehat{F}(\widehat{\mathbf{x}})$ and $\max\{F(\widehat{\mathbf{x}}), F(\widetilde{\mathbf{x}})\}$, respectively) and BMS (i.e., $\widetilde{F}(\widetilde{\mathbf{x}})$ and $F(\widetilde{\mathbf{x}})$, respectively), along with the optimal utility (i.e., $F(\mathbf{x}^*)$), for the different link densities and payoff distributions. The behavior of both algorithms is very similar across all link densities and payoff distributions. IBMS always computes an upper bound tighter than the one computed by BMS. The improvement is slightly better for the uniform distribution. The lower bounds computed by both algorithms are very close, although IBMS lower bound is slightly better.

Figure 2 (bottom row) shows a detail on the lower bounds $F(\widehat{\mathbf{x}})$ and $F(\widetilde{\mathbf{x}})$ obtained on each instance of a given parameter configuration. Since the behavior across all number of agents, link densities and payoff distributions is very similar, we only report results on instances with 25 agents and gamma distribution. Both lower bounds are very close, and none of them is consistently better than the other.

Figure 3 shows the percentage of improvement of the approximation ratio of IBMS $\rho$ and the weaker version of IBMS $\widehat{\rho}$ over the approximation ratio of MBS $\widetilde{\rho}$ (left y-axe). The figure also reports the percentage of deterioration of the approximation ratio of the 7-size-bounded-distance criteria introduced in [11] according to the minimum maximum reward bound ($S7_r$) and the minimum fraction bound ($S7_f$) presented in [12] over the approximation ratio of MBS $\widetilde{\rho}$ (right y-axe). Since the relation between the optimal solution of the problem $F(\mathbf{x}^*)$ and an approximation ratio $\rho$ of a given solution $\mathbf{x}$ is $1 \leq \frac{F(\mathbf{x}^*)}{F(\mathbf{x})} \leq \rho$, we compute the improvement of an approximation ratio $\rho$ over $\widetilde{\rho}$ as,

$$\frac{(\widetilde{\rho} - 1) - (\rho - 1)}{\widetilde{\rho} - 1} * 100$$

The improvement of $\rho$ is always higher than $37\%$, and up to almost $50\%$. Its mean improvement for the gamma and uniform distributions is higher than $40\%$ and $45\%$, respectively. The improvement of $\widehat{\rho}$ is always higher than $32\%$, and up to almost $46\%$. Its mean improvement for the gamma and uniform distributions is higher than $35\%$ and $37\%$, respectively. Therefore, both IBMS and its weaker version always significantly outperforms BMS. Recall that the weaker version of IBMS has the same communication demands as BMS. Both approximation ratios $S7_r$ and $S7_f$ are worse than the approximation ratio of BMS $\widetilde{\rho}$ (the percentage is always negative). Their quality decreases as the number of agents increases for both distributions.

## 6 Related Work

There are other two incomplete algorithms that can provide guarantees on the worst-case solution quality of their solutions at design time: *k-optimality* [8] and *t-optimality* [4]. The idea of these algorithms is to form coalitions of agents and to find the local optima solutions for all agents within the coalitions. This local optima is guaranteed to be within a predefined distance from the global optimal solution. Very recently, [11] proposed a framework were different coallition-based local optimality schemes can be described and defined a new criteria called *s-size bounded optimality*. The complexity of these algorithms depend on the number of coalitions and their size. Therefore, in practice, these algorithms are used with relatively small values of their control parameter.

In [10], it was shown that k-optimality provided significantly worst quality guarantees than BMS for different values of $k$. As stated in the following proposition, the quality guarantee provided by the 2-size-bounded optimality for binary DCOPs is always higher than 2.

**Proposition 1** *Let $P = (\mathbf{A}, \mathbf{X}, \mathbf{D}, \mathbf{F})$ be a binary DCOP (i.e., the arity of the functions is at most 2) such that $|\mathbf{X}| > 2$. The quality guarantee $\rho$ satisfies:*

$$2 \leq \rho \leq |\mathbf{F}|$$

*for all its 2-size-bounded optimal assignments.*

**Proof** According to [11],

$$\rho = \frac{|\mathcal{C}| - nc_*}{cc_*}$$

where:

- $\mathcal{C}$ is a multi-set of subsets of $\mathbf{X}$, where $C \in \mathcal{C}$ is a coalition;
- $cc_* = \min_{f \in \mathbf{F}}\{nc(f, \mathcal{C})\}$, where $cc(f, \mathcal{C}) = |\{C \in \mathcal{C} \mid var(f) \subseteq C\}|$;
- $nc_* = \min_{f \in \mathbf{F}}\{cc(f, \mathcal{C})\}$, where $nc(f, \mathcal{C}) = |\{C \in \mathcal{C} \mid var(f) \cap C = \emptyset\}|$;

Let $f_{ij}$ be a function with scope $\{x_i, x_j\}$. In a binary DCOP, its 2-size-bounded region is $\mathcal{C} = \{var(f) \mid f \in \mathbf{F}\}$ (i.e., $|\mathcal{C}| = |\mathbf{F}|$), $cc(f, \mathcal{C}) = 1$ for all $f \in \mathbf{F}$, and $nc(f_{ij}, \mathcal{C}) = |\mathbf{F}| - |\{f_{ik} \in \mathbf{F} \mid j \neq k\}| - |\{f_{lj} \in \mathbf{F} \mid l \neq i\}| + 1$ for all $f_{ij} \in \mathbf{F}$. The minimum $nc_*$ is when the DCOP constraint graph is a star because $nc_* = 0$, so that $\rho = |\mathcal{C}|$ is maximum. Note that on a star with $|\mathbf{X}| = 3$, $\rho = |\mathcal{C}| = 2$. The maximum $nc_*$ is when the DCOP constraint graph is a chain with $|\mathbf{X}| > 3$ (note that a chain with $|\mathbf{X}| = 3$ is a star), because $nc_* = |\mathbf{F}| - 3$ so that its $\rho = 3$. Note that a DCOP with two variables is trivially solved to optimality by the 2-bounded size optimality scheme.

Note that for all instances in the empirical evaluation, the quality guarantees of both IBMS and BMS were smaller than 1.4. For those instances, the s-size-bounded-distance provided worse approximation ratios than IBMS and BMS even using the improved minimum maximum reward and minimum fraction bounds and a relatively high value of the parameter $s$. We leave as future work a more extensive empirical comparison of the different algorithms.

## 7 Conclusions

In this paper we introduced a new algorithm, called Improved Bounded Max-Sum (IBMS), based on the Bounded Max-Sum algorithm. We theoretically proved that its approximation ratio is always better than the previous one, at the only cost of doubling the communication requirements. We also introduced a weaker version of IBMS having the same communication demands as Bounded Max-Sum. Our experiments show that the approximation ratio of both algorithms is significatively tighter.

## ACKNOWLEDGEMENTS

The authors are thankful to the authors of [10, 11] for providing us with their implementations, and to the authors of [10] for sharing their benchmarks with us. This work was supported by project TIN2009-13591-C02-01.

## REFERENCES

[1] S. M. Aji and R. J. McEliece, 'The generalized distributive law', *IEEE Transactions on Information Theory*, **46**(2), 325–343, (2000).
[2] A. Farinelli, A. Rogers, A. Petcu, and N. R. Jennings, 'Decentralised coordination of low-power embedded devices using the max-sum algorithm', in *AAMAS*, pp. 639–646, (2008).
[3] S. Fitzpatrick and L. Meetrens, 'Distributed coordination through anarchic optimization', in *Distributed Sensor Networks A multiagent perspective*, pp. 257–293. Kluwer Academic, (2003).
[4] C. Kiekintveld, Z. Yin, A. Kumar, and M. Tambe, 'Asynchronous algorithms for approximate distributed constraint optimization with quality bounds', in *AAMAS*, pp. 133–140, (2010).
[5] R. J. Maheswaran, J. Pearce, and M. Tambe, 'A family of graphical-game-based algorithms for distributed constraint optimization problems', in *Coordination of Large-Scale Multiagent Systems*, pp. 127–146. Springer-Verlag, (2005).
[6] R. Mailler and V. R. Lesser, 'Solving distributed constraint optimization problems using cooperative mediation', in *AAMAS*, pp. 438–445, (2004).
[7] P. J. Modi, W. M. Shen, M. Tambe, and M. Yokoo, 'Adopt: asynchronous distributed constraint optimization with quality guarantees', *Artif. Intell.*, **161**(1-2), 149–180, (2005).
[8] J. P. Pearce and M. Tambe, 'Quality guarantees on k-optimal solutions for distributed constraint optimization problems', in *IJCAI*, pp. 1446–1451, (2007).
[9] A. Petcu and B. Faltings, 'A scalable method for multiagent constraint optimization', in *IJCAI*, pp. 266–271, (2005).
[10] A. Rogers, A. Farinelli, R. Stranders, and N. R. Jennings, 'Bounded approximate decentralised coordination via the max-sum algorithm', *Artif. Intell.*, **175**(2), 730–759, (2011).
[11] M. Vinyals, E. Shieh, J. Cerquides, J. A. Rodriguez-Aguilar, Z. Yin, M. Tambe, and E. Bowring, 'Quality guarantees for region optimal dcop algorithms', in *AAMAS*, pp. 133–140, (2011).
[12] M. Vinyals, E. Shieh, J. Cerquides, J. A. Rodriguez-Aguilar, Z. Yin, M. Tambe, and E. Bowring, 'Reward-based region optimal quality guarantees', in *OPTMAS Workshop*, (2011).

# Computational issues surrounding the dynamic optimisation of management of an ecological food web

**William J M Probert** [1 2 3] and **Eve McDonald-Madden**[1] and **Nathalie Peyrard**[2] and **Régis Sabbadin**[2]

**Abstract.** We discuss computational issues surrounding current research that investigates the relevance of graph centrality metrics to the management of ecological food webs. Ecological food webs can be viewed as directed acyclic graphs and we use Markov decision processes to model management. Using dynamic programming we optimally solve the management of an Alaskan food web through time so as to maximise the expected number of species surviving. To generalise our results we investigate policies on generated food webs of varying size. For large food webs the state and action spaces are too large for dynamic programming to be computationally feasible and we use heuristical methods to approximate the optimal policy.

## 1 INTRODUCTION

Artificial intelligence has been proven to be a useful field for tackling the highly complex and uncertain nature of ecological systems [4, 20]. Management of such ecological systems, with the aim of preserving the world's natural resources in all forms, is the concern of conservation biology.

One area of ecological research in which artificial intelligence can contribute is in the application of graphical models to ecological food webs. In this context, ecological systems are represented mathematically as directed acyclic graphs. Nodes represent 'trophic species', which may be groups of plant or animal species, macroscopic or microscopic, that have a common set of predators and prey [26]. The term 'trophic' relates to the consumption of another species for energy. We will refer to these nodes of 'trophic species' as just 'species'. The basal species in a graph usually represent a ubiquitous food source such as zooplankton or kelp. Edges represent the trophic interactions between species. The terminating node of an edge represents the predator in the interaction (that is, the terminating node eats the source node).

In a reasonably general form, it is possible to frame the problem of managing a food web for conservation purposes as a Markov decision process (MDP). An MDP is an appealing choice because management is then explicitly part of the framework and solution algorithms exist [22]. The MDP framework has been used extensively within ecology and conservation [5, 21]. Solution policies, that prescribe an optimal action given a state and time step for a given reward structure, are specific to the structure of each food web in question and, due to the complexity of food webs, may only be solved using generous assumptions to simplify the system and available actions. Even then, solutions are only possible for moderately sized food webs of less than about 15 species. Such a constraint on finding exact solutions is considerable given that some documented food webs in the ecological literature have over 200 species groups (nodes), such as the Caribbean food web described in Bascompte et al (2005) [2]. It is therefore of interest to investigate heuristical policies that approximate the exact solution in small food webs so that approximate policies may be applied to manage larger food webs of more realistic sizes.

A myriad graph metrics have been applied to study ecological food webs and these provide a suitable starting point for investigating heuristic policies. Previous ecological and conservation research on food webs has mainly focused on measures of how the structure (trophic interactions between species) of a food web persists through species extinctions [15]. This 'stability' has numerous definitions, generally (and traditionally) referring to the stability of the characteristic polynomial from the square, real interaction matrix that defines the species interactions (as defined below) [9, 18].

Typically, the suggested metrics are calculated for each node and, thus, a ranking scheme of nodes is implicit. Research has yet to investigate the use and relevance of said metrics for the purposes of managing ecological systems [6, 1, 15, 13]. Our research compares management of a simple model of an ecological system using the optimal solution with management that is guided by ranking implied by graph metrics. If it is possible to identify metrics that provide management benefits (for a given reward function) that are close to the optimal solution for small scale food webs then such metrics may provide use in guiding management of larger networks.

Initial research has investigated the use of Bayesian networks to model this problem over one time step [19]. Bayesian networks are probability distributions with a structure defined on them [14, 16]. By 'structure', we mean that transition probabilities between different states can be factorised into a product of 'local' transition probabilities defined by the edges between nodes in a graph (and rewards may similarly be decomposed into a summation of local rewards). We extend the previous approach to the multiple time-step case using a finite-horizon Markov decision process (MDP).

An MDP is not just a Bayesian network that is extrapolated to the multiple time-step case and that includes modelling of decisions and rewards; this would usually be called an 'influence diagram' [12]. Figure 1 illustrates the relationship between different optimisation and modelling frameworks and the commonly used names for these techniques. The arrows indicate axes that add additional components to the framework: either time, structure or decisions and rewards. For instance, a Markov chain may be thought of as a one-dimensional random variable that includes a temporal component, similarly if one wishes to add a time component to a Bayesian network, the resultant

[1] ARC Centre for Excellence in Environmental Decisions (CEED), The University of Queensland, St Lucia, Queensland, 4072 Australia
[2] INRA-Toulouse, Unité de Biométrie et Intelligence Artificielle, Auzeville, Castanet-Tolosan, FRANCE
[3] email: wprobert AT maths DOT uq DOT edu DOT au; william DOT probert AT toulouse DOT inra DOT fr

model is typically called a dynamic Bayesian network, or if one has a one-dimensional random variable to which they wish to add a framework for decisions and rewards then the resultant framework is generally called stochastic optimisation. This diagram is only concerned with finite time horizons and the structure axis is not strictly one-way, in that, some of those techniques with structure may be modelled using those without formally defined structure (for instance, an influence diagram may be modelled as a Markov decision process).

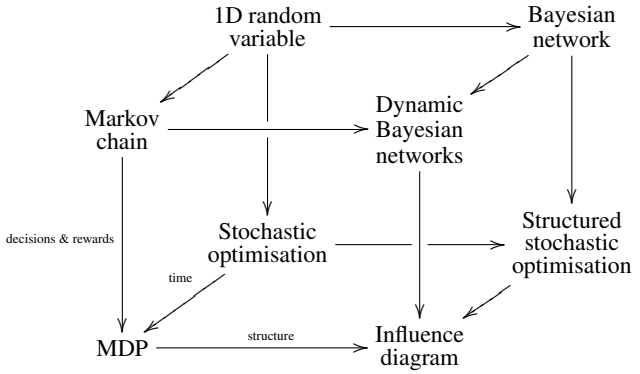



**Figure 1.** Relationships between modelling techniques with a finite time horizon

By modelling this decision making problem as an MDP, we risk facing the curse of dimensionality when trying to find an optimal solution; the state and action spaces can become combinatorially large as the number of species in the food web increases. Hence, it is necessary to investigate heuristical approaches to approximate exact solutions if results are to be extrapolated to food webs of realistic size. The exact solution and the heuristical solutions are compared via total expected rewards and using statistical analyses to search for patterns in the optimal policy. This is the first research to investigate the use of graph metrics for conservation management purposes over multiple time steps. The extension to multiple time steps also allows comparison of adaptive policies with the myopic policies of the graph heuristics. This manuscript focuses on the computational issues in calculating the exact solution algorithm and therefore some details of the current analysis are omitted.

## 2 FRAMEWORK & METHODS

We briefly outline the model used and then the computational issues and suggested work-arounds.

We have a directed acyclic graph, $G = \langle V, E \rangle$, representing a food web (figure 2). $V$ respresents a set of species (vertices) and $E$ represents a set of trophic interactions (oriented edges) between those species. There are $n$ species in our food web, $\mid V \mid = n$. Species $i$ is a prey of (is eaten by) species $j$ if there is an edge from $i$ to $j$, $(i, j) \in E$. We denote the adjacency matrix defining the food web with $\mathbf{G}$, where $\mathbf{G}_{i,j} = 1$ if $(i, j) \in E$, otherwise 0.

**Markov decision process.** A Markov decision process (MDP) may be used to model the decision making problem of managing an ecological food web. An MDP is a common framework for sequential decision making problems and is composed of a tuple, $\langle \mathcal{X}, \mathcal{A}, P, R \rangle$, consisting of the state space, action space, state transition probabilities and rewards, respectively [22]. We model this system for a set of time steps, $t \in \{1, \ldots, T\}$, where we have a finite time horizon, $T < \infty$. A superscript of $t$ represents the state or action at time step $t$. The state of each species can take one of two values, extant (1) or extinct (0), $x_i^t \in \{0, 1\},\ \forall i \in V$. The set $x^t$ then represents the state of all species in the food web at time $t$, $x^t \in \{0, 1\}^n$. Management actions that represent a species' protection status are available for each species. Simply, a species may or may not be protected, that is, $a_i^t \in \{0, 1\},\ \forall i \in V$, where protection takes the value 1. The term $a^t$ then represents the set of actions chosen for all species in the food web at time $t$, $a^t \in \{0, 1\}^n$. To recognise limitations in conservation funding, we introduce a budget constraint on the choice of actions at each time step, $c(a^t) = \sum c_i a_i^t \leq B^t$, for a per-time-step budget $B^t$, summation is over $i \in V$, and for a cost of protecting species $i$ of $c_i$ ($i \in V$).

To incorporate local interactions and dependencies in the food web we define neighbourhood functions. The assumption of a neighbourhood allows the factorisation of state transition probabilities into the product of species' local state transitions (and the respective decomposition of rewards into a sum of local rewards) [23]. We define a species' neighbourhood to include all prey species and the species itself, $N(i) = \{j \in V \mid (j, i) \in E\} \cup \{i\}$. The set $x_{N(i)}^t$ then denotes the states of all species in the set $N(i)$ at time $t$.

The transition probability function is defined as follows:

$$P^t(x^{t+1} \mid x^t, a^t) = \prod_{i=1}^{n} P_i^t(x_i^{t+1} \mid x_{N(i)}^t, a_i^t). \tag{1}$$

For simplicity, we assume transition probabilities are the same across time, $P_i^t(\cdot) = P_i(\cdot) \quad \forall t$. We define individual species' transition probabilities from one state to the next, for a given action, as some baseline probability of survival, $p_i^0$, times the ratio of the number of prey species that are extant ($f^{\star,t}$) to the total number of prey species ($f$),

$$P_i(x_i^{t+1} = 1 \mid x_i^t = 1, x_{N(i)\setminus i}, a_i^t = 0) = p_i^0 \left( \frac{f^{\star,t}}{f} \right)$$
$$P_i(x_i^{t+1} = 0 \mid x_i^t = 1, x_{N(i)\setminus i}, a_i^t = 0) = 1 - p_i^0 \left( \frac{f^{\star,t}}{f} \right)$$

Such a probability is subject to the following conditions:

- For basal species, the probability of transitioning from extant to extinct is solely equal to some baseline probability of survival, $p_i^0$.
- For any species, extinction (death) is an absorbing state,

$$P_i(x_i^{t+1} = 0 \mid x_i^t = 0, a_i^t) = 1, \forall a_i^t$$

- To survive, a species must have at least one prey species extant,

$$P_i(x_i^{t+1} = 0 \mid \sum_{k \in N(i)} x_k^t < 2, a_i^t) = 1, \forall a_i^t$$

- A species' is guaranteed to survive if it is protected, $a_i^t = 1$, and the above conditions hold,

$$P_i(x_i^{t+1} = 1 \mid x_i^t = 1, \sum_{k \in N(i)} x_k^t \geq 2, a_i^t = 1) = 1.$$

Transition probabilities are simplistic in this research project to avoid complexity in computation. Species demographics are not directly taken into account but they may be used in calculations to contribute to an overall probability of species extinction. Note that although basal species may be secured indefinitely by protecting them, this

does not guarantee that species further up the food chain will not go extinct and thus may not lead to the most desirable outcome for a given choice of reward function.

We denote the array of transition probabilities as $\mathbf{P}$, where $\mathbf{P}_{i,i',a}$ is the probability of transitioning from state $i$ to state $i'$ when action $a$ is taken.

Our objective is to maximise the number of species surviving at the end of the project. Rewards are assumed stationary through time (with the exception of rewards in the final time step). We use a final time-step reward function of the number of extant species,

$$R^T(x^T) = \sum_{i=1}^{n} x_i^T \tag{2}$$

and per-time-step rewards are zero, $R^t(x^t) = 0,\ t < T$. Various different reward functions may be investigated, including rewards that acknowledge the presence of ecologically meaningful structure in the state of the food web, but for conciseness we only mention one basic reward function here.

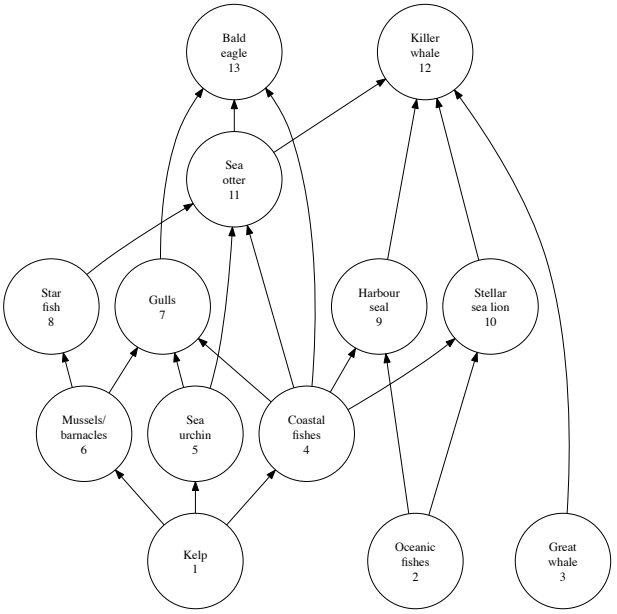



**Figure 2.** Example food web based on an Alaskan trophic network with 13 species and 21 trophic edges. The numbers underneath the names of the species represent the species index, $i$, which are referred to in the subsequent tables.

**Policies.** Let $\delta_r = (d^1, \ldots, d^{T-1})$ represent a policy according to rule $r$, that designates an action, $a^t = d^t(x^t)$, to take at each time step according to the current state, $x^t$, and decision rule $d^t(x^t)$. The total expected reward of any policy, $\delta$, is defined as

$$v_\delta^T(x^1) = \mathrm{E}\left[\sum_{t=1}^{T} R^t(x^t) \,\middle|\, x^1, \delta\right] \tag{3}$$

**Optimal policy.** Solving the above MDP involves finding the optimal policy, $\delta_*$, that provides the highest total expected reward. This maximum total expected reward is called the 'value'. For food webs with 13 species or less, we may solve the above MDP using the backwards induction algorithm as follows [22]:

1. Set the current time-step to $t = T$ and the value in the final time-step to $v_*^T(x^T) = R^T(x^T)\ \forall x^T \in \mathcal{X}$
2. Set $t = t - 1$ and calculate $v_*^t(x^t)$ for each state using

$$v_*^t(x^t) = \max_{a^t \in \mathcal{A}} Q^t(x^t, a^t)$$
$$a_*^t = \arg\max_{a^t \in \mathcal{A}} Q^t(x^t, a^t)$$

where

$$Q^t(x^t, a^t) = R^t(x^t) + \sum_{x^{t+1}} P(x^{t+1} \mid x^t, a^t)\ v_*^{t+1}(x^{t+1})$$

3. If $t = 1$ then stop, otherwise return to step 2.

For our problem, this is initialised by setting the optimal value in the final time step equal to the final rewards in equation (3) and setting per-time step rewards to zero.

**Graph based policies.** We generate policies guided by ranking schemes that are defined by several graph metrics, outlined below. Graph metrics map nodes to integer or real values. Policies that are defined on graph metrics will manage nodes in descending order of the graph metric. Ties in graph metrics are determined by randomisation and isolated species, for instance if the current state causes the food web to become disconnected, have metric values of zero (which means they are managed last).

The total expected rewards of a graph based policy is evaluated using the following finite horizon metric-policy evaluation algorithm with inputs $\delta = (d^1, \cdots, d^{T-1})$ and the interaction matrix, $\mathbf{G}$ [22]:

1. Set the current time-step to $t = T$ and the terminal rewards in the final time-step to $v_\delta^T(x^T) = R^T(x^T)\ \forall\ x^T \in \mathcal{X}$
2. Set $t = t - 1$ and calculate $v_\delta^t(x^t)$ for each state using

$$v_\delta^t(x^t) = Q^t(x^t, d(x^t))$$

where

$$Q^t(x^t, d(x^t)) = R^t(x^t) \quad + \sum_{x^{t+1}} P(x^{t+1} \mid x^t, d^t(x^t))\ v_\delta^{t+1}(x^{t+1}) \tag{4}$$

3. If $t = 1$ then stop, otherwise return to step 2.

This algorithm is also initialised by setting final time step rewards equal to equation (3) and setting per-time step rewards to zero.

Below we define several graph metrics. From the social or mathematical sciences we define degree centrality, betweenness centrality and closeness centrality. From the ecological literature we define prey degree, predator degree, keystone index, bottom-up prioritisation and trophic level. Metric values for each species in the 13 species Alaskan food web of figure 2 are presented in tables 3 and 4.

*Degree centrality*

The degree of a species is the number of connections (in any direction) to a particular species [25]. This can be normalised by the size of the food web so that degree centrality can be compared between graphs of different sizes (note that $n - 1$ is the maximum number of

connections a species can have in any graph). The degree of species $i$ is the sum of the number of prey and predators that it has, normalised by the size of the food web,

$$D_i = \frac{D_i^{\leftarrow} + D_i^{\rightarrow}}{\mid V \mid -1} \tag{5}$$

where $D_i^{\leftarrow}$ and $D_i^{\rightarrow}$ are respectively the prey degree and predator degree of species $i$. More specifically, $D_i^{\leftarrow}$ is the size of the set $V_i^{\leftarrow} = \{j \in V : (j,i) \in E\}$ of all prey of species $i$, and $D_i^{\rightarrow}$ is the size of the set $V_i^{\rightarrow} = \{j \in V : (i,j) \in E\}$, the set of all predators of species $i$.

*Betweenness centrality*

The betweenness centrality of a species reflects how central a species is in the transmission of energy that links other species in the food web. It is the proportion of shortest paths between any two species that pass through a particular species, normalised by the size of a network [25]. Betweenness centrality is calculated as

$$BC_i = \frac{\sum\limits_{j<k} \frac{g_{jk}(i)}{g_{jk}}}{(\mid V \mid -1)(\mid V \mid -2)}, \tag{6}$$

where $g_{jk}$ denotes the number of shortest paths (geodesics) between species $j$ and $k$, and $g_{jk}(i)$ denotes the number of shortest paths between species $j$ and species $k$ which pass through species $i$.

*Closeness centrality*

Closeness centrality is a function of the sum of distances between a node and all other nodes in the graph [3, 7, 8]. Specifically it is the ratio of 1) the sum of distances from a particular species to every other species in the network to 2) the minimum possible value that this sum may take (which is $n-1$ for a network of size $n$).

The closeness centrality for species $k$ is

$$CC_k = \left[ \frac{\sum\limits_{i=1}^{n} d(i,k)}{n-1} \right]^{-1} \tag{7}$$

This is a relative measure and can be compared between networks of different sizes. The distance between two species, $d(i,j)$, is the smallest number of trophic connections between species $i$ and $j$ (geodesic). Note that the sum of distances from a species to all other species will grow with the distance between species (hence the inverse is taken) and distances ignore direction of the trophic interaction.

*Keystone index*

The keystone index is based on the idea of 'status' and 'contrastatus' of an organisation by Hararay [10, 11, 15]. This measure is only for directed, acyclic graphs. Using the definitions of $V_i^{\rightarrow}$ and $V_i^{\leftarrow}$ from the degree centrality description the keystone index may be composed of two sets of two components:

$$\begin{aligned} K_i &= K_i^{\downarrow} + K_i^{\uparrow} \\ K_i^{\downarrow} &= \sum_{c \in V_i^{\leftarrow}} \frac{1}{D_c^{\leftarrow}} (1 + K_c^{\downarrow}) \\ K_i^{\uparrow} &= \sum_{e \in V_i^{\rightarrow}} \frac{1}{D_e^{\rightarrow}} (1 + K_e^{\uparrow}) \end{aligned} \tag{8}$$

| $i$ | $D_i$ | $D_i^{\leftarrow}$ | $D_i^{\rightarrow}$ | $BC_i$ | $CC_i$ |
|---|---|---|---|---|---|
| 1 | 3 | 0 | 3 | 0 | 0.48 |
| 2 | 2 | 0 | 2 | 0 | 0.364 |
| 3 | 1 | 0 | 1 | 0 | 0.353 |
| 4 | 6 | 1 | 5 | 0.035 | 0.632 |
| 5 | 3 | 1 | 2 | 0.008 | 0.462 |
| 6 | 3 | 1 | 2 | 0.01 | 0.429 |
| 7 | 4 | 3 | 1 | 0.011 | 0.5 |
| 8 | 2 | 1 | 1 | 0.015 | 0.444 |
| 9 | 3 | 2 | 1 | 0.008 | 0.5 |
| 10 | 3 | 2 | 1 | 0.008 | 0.5 |
| 11 | 5 | 3 | 2 | 0.04 | 0.6 |
| 12 | 4 | 4 | 0 | 0 | 0.522 |
| 13 | 3 | 3 | 0 | 0 | 0.522 |
| Mean | 3.231 | 1.615 | 1.615 | 0.01 | 0.485 |
| SD | 1.25 | 1.273 | 1.273 | 0.013 | 0.076 |
| Min | 1 | 0 | 0 | 0 | 0.353 |
| Max | 6 | 4 | 5 | 0.04 | 0.632 |

**Figure 3.** Graph metrics for each species in the full Alaskan food web of figure 2. Metrics are respectively (L to R in columns 2 to 6) degree, prey degree, predator degree, betweenness centrality, and closeness centrality. Index numbers, $i$, for each species are labelled in figure 2.

where $K_i^{\downarrow}$ and $K_i^{\uparrow}$ represent the top-down and bottom-up keystone indices of node $i$ respectively. Alternatively, equation (8) can be rearranged to express the keystone index as a sum of direct and indirect effects on node $i$,

$$\begin{aligned} K_i &= K_i^{\text{dir}} + K_i^{\text{undir}} \\ K_i^{\text{dir}} &= \sum_{c \in V_i^{\leftarrow}} \frac{1}{D_c^{\leftarrow}} + \sum_{e \in V_i^{\rightarrow}} \frac{1}{D_e^{\rightarrow}} \\ K_i^{\text{undir}} &= \sum_{c \in V_i^{\leftarrow}} \frac{K_c^{\uparrow}}{D_c^{\leftarrow}} + \sum_{e \in V_i^{\rightarrow}} \frac{K_e^{\downarrow}}{D_e^{\rightarrow}} \end{aligned}$$

*Bottom-up prioritisation*

Bottom-up prioritisation (BUP) ranks species firstly according to trophic level, $L_i$, which is calculated from the complete food web, and then secondly by the number of extant predators in the current state. Let $L_1$ define the set of basal species in a food web with a trophic level of 1, that is, all those species which do not have prey in the complete food web,

$$L_1 = \{k \in V : D_k^{\leftarrow} = 0\} \tag{9}$$

We then define the subgraph food web, $G_i = (V_i, E_i)$, as that which excludes species in trophic level $i$ and all trophic levels below $i$, where

$$V_i = V \setminus \bigcup_{k<i} L_k \tag{10}$$

and $E_i$ is the corresponding edges between these vertices from the original graph. All species with trophic level $i$ are then those which belong to the set

$$L_i = \left\{ k \in V_i : D_k^{\leftarrow,i} = 0 \right\} \tag{11}$$

| $i$ | $K_i$ | $K_i^{\text{dir}}$ | $K_i^{\text{indir}}$ | $K_i^{\text{bu}}$ | $K_i^{\text{td}}$ | $L_i$ |
|---|---|---|---|---|---|---|
| 1 | 8.5 | 3 | 5.5 | 8.5 | 0 | 1 |
| 2 | 1.25 | 1 | 0.25 | 1.25 | 0 | 1 |
| 3 | 0.25 | 0.25 | 0 | 0.25 | 0 | 1 |
| 4 | 2.889 | 2.333 | 0.556 | 2.556 | 0.333 | 2 |
| 5 | 1.306 | 1 | 0.306 | 0.972 | 0.333 | 2 |
| 6 | 2.306 | 1.667 | 0.639 | 1.972 | 0.333 | 2 |
| 7 | 1.933 | 1.533 | 0.4 | 0.333 | 1.6 | 3 |
| 8 | 1.194 | 0.833 | 0.361 | 0.528 | 0.667 | 3 |
| 9 | 1.017 | 0.95 | 0.067 | 0.25 | 0.767 | 3 |
| 10 | 1.017 | 0.95 | 0.067 | 0.25 | 0.767 | 3 |
| 11 | 3.183 | 2.283 | 0.9 | 0.583 | 2.6 | 4 |
| 12 | 6.333 | 3.5 | 2.833 | 0 | 6.333 | 5 |
| 13 | 4.667 | 1.7 | 2.967 | 0 | 4.667 | 5 |
| Mean | 2.757 | 1.615 | 1.142 | 1.342 | 1.415 | 2.692 |
| SD | 2.315 | 0.902 | 1.572 | 2.196 | 1.904 | 1.323 |
| Min | 0.25 | 0.25 | 0 | 0 | 0 | 1 |
| Max | 8.5 | 3.5 | 5.5 | 8.5 | 6.333 | 5 |

**Figure 4.** Graph metrics for each species in the full Alaskan food web of figure 2. Metrics are respectively (L to R in columns 2 to 7) keystone index, directed keystone index, indirected keystone index, bottom-up keystone index, top-down keystone index and trophic level. Index numbers, $i$, for each species are drawn in figure 2.

where $D_k^{\leftarrow,i}$ is the prey degree of species $k$ in the food web defined by $G_i$. In other words, a species is in trophic level 3, for example, if it has no prey species present after removing species in trophic levels 1 and 2. The bottom-up prioritisation ranks species first by ascending trophic level, and then by descending number of predators.

**Policies and comparisons.** In addition to the policies defined by graph metrics and the optimal policy, we also include the policy that manages nothing and a policy that chooses extant species to protect at random. The total expected reward of policies, calculated from the metric-policy evaluation algorithm, will be compared with the optimal value (the maximum total expected reward) from the exact solution, using the backwards induction algorithm, to find which approximation methods are the best.

**Computational issues and remedies.** To utilise the backwards induction algorithm, transition probabilities must be calculated for the current problem. The number of probabilities to calculate will be $\mid S \mid \times \mid S \mid \times \mid A \mid$. For a small real food web with 13 species and a budget with the capacity to protect 4 species at each time step, we have $2^{13}.2^{13}.C_4^{13} = 47,982,837,760$ transition probabilities to calculate (over 40 billion). With 25 species and a budget capacity to protect 8 species, this is more than $1.2 \times 10^{21}$ transition probabilities, illustrating the need to use approximations to the optimal dynamic programming solution when dealing with larger food webs. Our modelling framework already uses generous assumptions to simplify the model and uses transition probabilities that do not change through time. Previously studied food webs have included over 150 different groups of organisms [17, 2] which is out of reach for the optimal solution in the current investigation.

Figure 5 plots the average CPU time over 10 different food webs for calculating four metrics against the food web size, of up to 25 species. Computations were performed using Matlab version 7.7.0.471 (Mathworks, 2009). A selection of 10 food webs of each size were randomly generated using a published method for generating food webs, the cascade model, and using a 'connectance' value, $C = |E|/|V|^2$, of 0.1 [24, 26]. A cascade model is one which is constructed by first assigning each species a uniform random number and secondly setting the probability that a species predates on species with a random uniform value less than its own to $2.C.n/(n-1)$ [24, 26]. Connectance values in real, observed food webs range between 0.1 and 0.2 [17, 26]. Calculations were performed under MacOS 10.6.8 on a 2.53GhZ, mid-2009 Macbook Pro with 4Gb of 1067MHz DDR3 RAM.

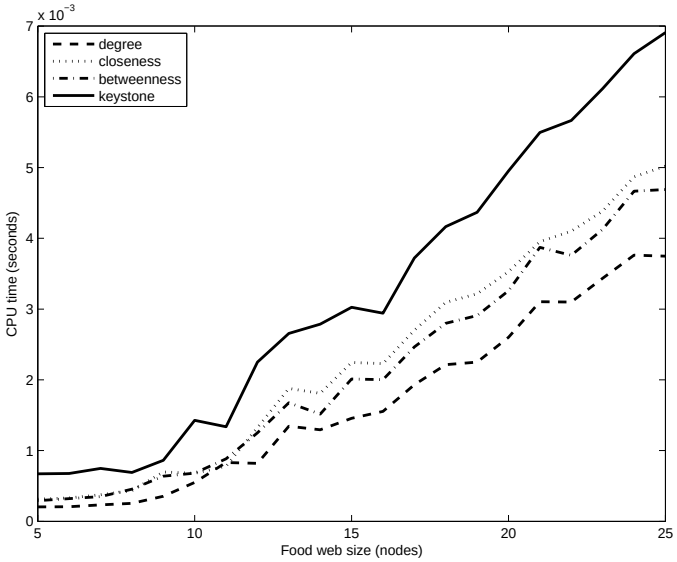



**Figure 5.** Average CPU time for calculation of four different metrics as a function of food web size.

A couple of steps can be used to remedy the curse of dimensionality. As mentioned, we assumed transition probabilities may be factorised into a product of species' local transitions. This means it is only necessary to calculate the probability of all possible local transitions for any species. Furthermore, various transitions in states can be set to zero based on the conditions of the transition probabilities. Concretely, consider the matrix $\mathbf{M} = \mathbf{SG}$, where $\mathbf{S} : \mathcal{X} \rightarrow 2^{\mathcal{X}}$, is a $2^n \times n$ Boolean matrix that indicates for each possible state which species is extant (one row for each state, columns index species) and $\mathbf{G}$ is the $n \times n$ adjacency matrix of the food web in question. That is, $\mathbf{G}_{ij} = 1$ if species $i$ is a prey of species $j$ and otherwise 0; cannibalism is not allowed $\mathbf{G}_{ii} = 0 \; \forall \; i$. The elements of the $2^n \times n$ matrix $\mathbf{M}$ are then

$$\mathbf{M}_{i,j} = \text{Number of extant prey of species } j \text{ when the state is } \mathbf{S}_{i,:}$$

where $\mathbf{S}_{i,:}$ is the $i$th row of $\mathbf{S}$. It is not necessary for basal species to have prey to survive so we set $M_{i,j} = 1$ if species $j$ is a basal species. Further, defining $\mathbf{Q} = \mathbf{M} \odot \mathbf{S}$, where $\odot$ defines element-wise multiplication (the Hadamard product), we have a matrix that has elements

$$\mathbf{Q}_{i,j} = \begin{cases} 0 & \text{if species } j \text{ is extinct or has no prey in state } i \\ & \text{(and is not a basal species)} \\ \{1, 2, \cdots\} & \text{otherwise} \end{cases}$$

and thus $\mathbf{P}_{i,i',a} = 0$ if $\mathbf{Q}_{i,j} = 0 \; \forall j$ s.t. $\mathbf{S}_{i',j} > 0$.

## 3 Results

Despite the methods described above for speeding up computation time, the calculation of transition probabilities still takes a considerable amount of time. For illustrative purposes, we present initial results (table 6) for the Alaskan food web with only 10 species (by removing the Great whale, star fish and mussels/barnacles from the 13 species food web). It is possible to solve the exact solution to the 10 species web in several minutes on the computer described above.

| Policy | $\upsilon_\delta(x^1)$ |
|---|---|
| Optimal | 5.92 |
| $K^{\uparrow}$ | 5.52 |
| BUP | 5.52 |
| $D^{\rightarrow}$ | 5.51 |
| $K$ | 4.99 |
| $K^{\text{indir}}$ | 4.97 |
| $K^{\text{dir}}$ | 4.76 |
| $BC$ | 4.00 |
| $D$ | 3.84 |
| $CC$ | 3.72 |
| Random | 3.66 |
| $K^{\downarrow}$ | 3.49 |
| $D^{\leftarrow}$ | 3.45 |
| None | 1.10 |

**Figure 6.** Preliminary results for the 10 species Alaskan food web (with Great whale, star fish and mussels/barnacles removed) over 10 time steps with $p_i^0 = 0.9,\ \forall i$, and budget of 4 protected species at each time step. Column 2 is the total expected number of species surviving in the final time step when using each policy to manage the full 10 species Alaskan food web for 10 years (higher is better). Rows are arranged in descending order of their values in column 2.

We present the total expected reward (and respectively the value in the exact policy) in the first time step for the 10 species food web for a project of 10 years, with an underlying probability of survival of $p_i^0 = 0.9, \forall i$, and budget capacity to protect 4 species at each time step. Terminal rewards are the final number of species surviving and per-time-step rewards are zero. The policies presented are based on the eleven metrics described above, the random policy, the policy of protecting no species, and the optimal policy.

Results from the 10 species Alaskan food web suggest that the heuristic policies that perform well compared to the exact solution are those which are based on metrics that acknowledge the number of extant predators that a species has (the bottom-up keystone index, predator degree, and bottom-up prioritisation). On the other hand, heuristic policies that prioritise management of species based on the number of extant prey perform worse than a random strategy (prey degree, top-down keystone index).

## 4 Discussion

We have discussed both exact methods for solving a Markov decision processes with an underlying graphical structure and the potential of heuristics, based on graph metrics, for guiding decision making when exact methods are not computationally feasible. Preliminary results have been presented for a 10 species food web and suggest that heuristic policies that prioritise species according to the number of extant predators perform the best. Results suggest management preference for species of a lower trophic level which may be associated with the assumption of guaranteed survival for protected basal species.

Ten species is relatively small for an ecological food web and, thus, findings should be taken cautiously at this early stage of research. Future research will compare the exact solution with heuristic policies for additional food webs in the ecological literature and use statistical analyses to search for patterns in the optimal policies that may be predicted by either graph metrics or local features of the food web. Heuristic policies that are found to approximate the optimal solution consistently will be used to simulate management of larger food webs. Future research will also investigate the potential of more sophisticated methods of approximating the exact method to ecological food-web management, similar to those used in graph-based MDPs [23].

## ACKNOWLEDGEMENTS

Thanks to INRA-BIA in Toulouse for WP's travel and accommodation funding for the ECAI workshop, and thanks to the Australian Research Council's Centre of Excellence for Environmental Decisions for WP's funding to attend the workshop. We wish to acknowledge two reviewers for pertinent comments that improved this manuscript.

## REFERENCES

[1] M. Almeida-Neto, P. Puimarães, P. R. Puimarães Jr, R. D. Loyola, and W. Ulrich, 'A consistent metric for nestedness analysis in ecological systems: reconciling concept and measurement', *Oikos*, **117**, 1227–1239, (2008).

[2] J. Bascompte, C. J. Melián, and E. Sala, 'Interaction strength combinations and the overfishing of a marine food web', *Proceedings of the National Academy of Sciences of the United States of America*, **102**, 5443–5447, (2005).

[3] M. A. Beauchamp, 'An improved index of centrality', *Behavioral Science*, **10**(2), 161–163, (1965).

[4] I. Chadès, T.G. Martin, S. Nicol, M.A. Burgman, H.P. Possingham, and Y.M. Buckley, 'General rules for managing and surveying networks of pests, diseases, and endangered species', *Proceedings of the National Academy of Sciences of the United States of America*, **108**, 8323–8328, (2011).

[5] I. Chadès, E. McDonald-Madden, M.A. McCarthy, B. Wintle, M. Linkie, and H.P. Possingham, 'When to stop managing or surveying cryptic threatened species', *PNAS*, **105**(37), 13936–13940, (2008).

[6] E. Estrada, 'Characterization of topological keystone species: local, global, and 'meso-scale' centralities in food webs', *Ecological Complexity*, **4**, 48–57, (2007).

[7] L. C. Freeman, 'Centrality in social networks: conceptual clarification', *Social Networks*, **1**, 215–239, (1978).

[8] L. C. Freeman, 'The gatekeeper, pair-dependency and structural centrality', *Quality and Quantity*, **14**, 585–592, (1980).

[9] M. R. Gardner and W. R. Ashby, 'Connectance of large dynamic (cybernetic) systems: critical values for stability', *Nature*, **228**, 784, (1970).

[10] F. Harary, 'Status and contrastatus', *Sociometry*, **22**(1), 23–43, (1959).

[11] F. Harary, 'A structural analysis of the situation in the Middle East in 1956', *The Journal of conflict resolution*, **5**(2), 167–178, (1961).

[12] R.A. Howard and J.E. Matheson, 'Influence diagrams', in *Readings on the Principles and Applications of Decision Analysis, Vol. II*, eds., R.A. Howard and J.E. Matheson, Menlo Park CA: Strategic Decisions Group, (1984).

[13] A. James, J. W. Pitchford, and M. J. Plank, 'Disentangling nestedness from models of ecological complexity', *Nature*, **487**, 227–230, (2012).

[14] F. V. Jensen and T. D. Nielsen, *Bayesian Networks and Decision Graphs*, Springer Verlag, 2001.

[15] F. Jordán, 'Keystone species and food webs', *Philosophical Transactions of The Royal Society, B*, **364**, 1733–1741, (2009).

[16] D. Koller and N. Friedman, *Probabilistic graphical models: principles and techniques*, MIT Press, 2009.

[17] N. D. Martinez, 'Artifacts or attributes? effects of resolution on the Little Rock Lake food web', *Ecological Monographs*, **61**, 367–392, (1991).
[18] R. May, 'Will a large complex system be stable?', *Nature*, **238**, 413–414, (1972).
[19] R. McDonald-Madden, E.and Sabbadin, P. W. J. Baxter, I. Chadès, and H. P. Possingham, 'Making food web theory useful for conservation decisions', *Submitted*, (2013).
[20] S.C. Nicol, I. Chadès, S. Linke, and H.P. Possingham, 'Conservation decision-making in large state spaces', *Ecological Modelling*, **221**, 2531–2536, (2010).
[21] W. J. M. Probert, C. E. Hauser, E. McDonald-Madden, M. C. Runge, P. W. J. Baxter, and H. P. Possingham, 'Managing and learning with multiple models: objectives and optimization algorithms', *Biological Conservation*, **144**, 1237–1245, (2011).
[22] M. L. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming*, John Wiley and Sons, New York, NY, 1994.
[23] R. Sabbadin, N. Peyrard, and N. Forsell, 'A framework and a mean-field algorithm for the local control of spatial processes', *International Journal of Approximate Reasoning*, **53**, 66–86, (2012).
[24] A. R. Solow and A. R. Beet, 'On lumping species in food webs', *Ecology*, **79**(6), 2013–2018, (1998).
[25] S. Wasserman and K. Faust, *Social Network Analysis*, Cambridge University Press, Cambridge, 1994.
[26] R. J. Williams and N. D. Martinez, 'Simple rules yield complex food webs', *Nature*, **404**, 180–183, (2000).

# An upper bound for BDeu local scores

**James Cussens**[1]

**Abstract.** An upper bound on BDeu log local scores is derived using an existing upper bound on the beta function with $r$ variables. Two bounds on this bound are derived, one of which is suitable for pruning the search for optimal parent sets of a variable in Bayesian network learning. Empirical results concerning the tightness of bounds are given.

## 1 BDeu log local scores

If Dirichlet parameters are used for the parameters of a BN and the data $D$ is complete then the log marginal likelihood for BN structure $G$ with variables $i = 1, \ldots, p$ is:

$$\log P(G|D) = \sum_{i=1}^{p} z_i(G)$$

where

$$z_i(G) = \sum_{j=1}^{q_i(G)} \left( \log \frac{\Gamma(\alpha_{ij})}{\Gamma(n_{ij} + \alpha_{ij})} + \sum_{k=1}^{r_i} \log \frac{\Gamma(n_{ijk} + \alpha_{ijk})}{\Gamma(\alpha_{ijk})} \right) \tag{1}$$

Defining notation in (1): $q_i(G)$ is the number of joint instantiations of the parents of $i$ in $G$; $r_i$ is the number of values variable $i$ can take; $n_{ijk}$ is the count of how often in the data variable $i$ takes its $k$th value when its parents in $G$ take their $j$th joint instantiation; and $\alpha_{ijk}$ is the Dirichlet parameter corresponding to $n_{ijk}$. We also have $n_{ij} = \sum_{k=1}^{r_i} n_{ijk}$ and $\alpha_{ij} = \sum_{k=1}^{r_i} \alpha_{ijk}$. This equation (in non-log form) is given in [7] and was first derived by Cooper and Herskovits [2]. In this paper the parameters $\alpha_{ijk}$ will be set to $\alpha/(r_i q_i(G))$ where $\alpha$ (known as the *equivalent sample size*) is set by the user. This variant is known as the log BDeu score (Bayesian Dirichlet equivalent uniform score) where the "uniform" reflects that $\alpha_{ijk}$ is the same for all values $k$ of $i$. With this restriction, and abbreviating $q_i(G)$ to $q_i$, (1) becomes:

$$z_i(G) = \sum_{j=1}^{q_i} \left( \log \frac{\Gamma(\frac{\alpha}{q_i})}{\Gamma(n_{ij} + \frac{\alpha}{q_i})} + \sum_{k=1}^{r_i} \log \frac{\Gamma(n_{ijk} + \frac{\alpha}{q_i r_i})}{\Gamma(\frac{\alpha}{q_i r_i})} \right) \tag{2}$$

$z_i(G)$ is the *BDeu log local score* for variable $i$ and is determined by the parents $i$ has in $G$ (as well as by the data). Note that if $n_{ij} = 0$ for some $j$ then also $n_{ijk} = 0$ for all $k$ and the summand for $j$ in (2) is zero. Let $q^{(0)}$ be the number of values of $j$ where $n_{ij} = 0$ and $q^{(+)}$ be the number of values of $j$ where $n_{ij} \geq 1$ (so that $q_i = q^{(0)} + q^{(+)}$). Suppose, without loss of generality, that $j = 1, \ldots q^{(+)}$ are the values of $j$ where $n_{ij} \geq 1$, then we have that:

$$z_i(G) = \sum_{j=1}^{q^{(+)}} \left( \log \frac{\Gamma(\frac{\alpha}{q_i})}{\Gamma(n_{ij} + \frac{\alpha}{q_i})} + \sum_{k=1}^{r_i} \log \frac{\Gamma(n_{ijk} + \frac{\alpha}{q_i r_i})}{\Gamma(\frac{\alpha}{q_i r_i})} \right) \tag{3}$$

Since the log BDeu score exclusively is used here *BDeu log local score* will be abbreviated to *local score*. A more compact formulation of (3) is possible using the beta function of $r$ variables. This function $B(x_1, \ldots, x_r)$ is defined in (4).

$$B(x_1, \ldots, x_r) = \frac{\Gamma(x_1) \ldots \Gamma(x_r)}{\Gamma(x_1 + \ldots + x_r)} \tag{4}$$

so that

$$\log B(x_1, \ldots, x_r) = \left( \sum_{k=1}^{r} \log \Gamma(x_k) \right) - \log \Gamma \left( \sum_{k=1}^{r} x_k \right)$$

Noting that

$$\sum_{k=1}^{r_i} \left( n_{ijk} + \frac{\alpha}{q_i r_i} \right) = n_{ij} + \frac{\alpha}{q_i}$$

we have

$$\begin{aligned} z_i(G) &= \sum_{j=1}^{q^{(+)}} \left( \log \frac{\Gamma(\frac{\alpha}{q_i})}{\Gamma(n_{ij} + \frac{\alpha}{q_i})} + \sum_{k=1}^{r_i} \log \frac{\Gamma(n_{ijk} + \frac{\alpha}{r_i q_i})}{\Gamma(\frac{\alpha}{q_i r_i})} \right) \\ &= \sum_{j=1}^{q^{(+)}} \left[ \log \Gamma \left( \frac{\alpha}{q_i} \right) - \left( \sum_{k=1}^{r_i} \log \Gamma \left( \frac{\alpha}{q_i r_i} \right) \right) \right. \\ &\quad \left. + \left( \sum_{k=1}^{r_i} \log \Gamma \left( n_{ijk} + \frac{\alpha}{q_i r_i} \right) \right) - \log \Gamma \left( n_{ij} + \frac{\alpha}{q_i} \right) \right] \\ &= q^{(+)} \log \Gamma \left( \frac{\alpha}{q_i} \right) - q^{(+)} r_i \log \Gamma \left( \frac{\alpha}{q_i r_i} \right) \\ &\quad + \sum_{j=1}^{q^{(+)}} \log B \left( n_{ij1} + \frac{\alpha}{q_i r_i}, \ldots n_{ijr_i} + \frac{\alpha}{q_i r_i} \right) \end{aligned} \tag{5}$$

## 2 Computational benefits of bounding BDeu scores

An important problem is to identify a BN structure with maximal BDeu score for a given data set. Upper bounds on local scores can help in this task. To see this first note that the local score $z_i(G)$ depends only on the parents that $i$ has in the graph $G$ (that is what makes it 'local'), so it is clearer to write the local score as $z_i(W)$ where $W \subseteq \{1, \ldots p\} \setminus \{i\}$ are the parents of $i$. We have the following useful result:

[1] Dept. of Computer Science & York Centre for Complex Systems Analysis, University of York, York, UK email:james.cussens@york.ac.uk

**Theorem 1** *If $W \subset W'$ and $z_i(W) > z_i(W')$ then $W'$ cannot be a parent set for $i$ in an optimal BN.*

This is because any BN where $W'$ were the parents of $i$ would have a worse score than one where the edges from $W' \setminus W$ to $i$ were removed.

This simple result has been used by many working on BN learning [8, 3, 5, 4], but it has been exploited most fruitfully by de Campos and Ji [6]. Let $q_i(W')$ be the number of joint instantiations of variables in the set $W'$, and let $q^{(+)}(W')$ be the number of associated non-zero counts. Translating the results of de Campos and Ji to the notation of the current paper they show that if (i) $W \subset W'$ and (ii) $\alpha/q_i(W') \leq 0.8349$ and (iii) $z_i(W) > -q^{(+)}(W') \log r_i$ then neither $W'$ *nor and any superset of* $W'$ can be an optimal parent set for $i$. (Their result is actually more general since they consider any BDe score, not just as here, the BDeu score.)

This is a tremendously useful result. Before computing the local score of a candidate parent set $W'$ for $i$, de Campos and Ji inspect the scores of any previously computed $z_i(W)$ where $W \subset W'$ to see if conditions (ii)-(iii) are satisfied. If so, there is no need to compute $z_i(W')$. More importantly, all supersets of $W'$ can be ruled out as optimal parent sets for $i$. Note that the required quantities for this check are either readily available or appropriately bounded: $\alpha$ is fixed by the user, $\log r_i$ requires a simple look-up, $q_i(W') \geq q_i(W)$ and $q^{(+)}(W') \geq q^{(+)}(W)$.

de Campos and Ji establish their result by considering upper bounds on BDe scores. The goals of the current paper are the same (restricted to BDeu). For an upper bound to be useful it important that (i) it allows all supersets of some $W'$ to ruled out as parents set for $i$ and (ii) it is defined in terms of cheaply computable quantities. The next section derives a promising upper bound for this.

## 3 Exploiting Alzer's bound

The key contribution of this paper is that we can obtain a useful upper bound on (BDeu log) local scores using an upper bound on the beta function discovered by Alzer [1]. Here is Alzer's result in its general form.

**Theorem 2** *(From [1]). Let $c > 0$ be a real number and let $r \geq 2$ be an integer. Then we have for all real numbers $x_k \geq c\,(k = 1, \ldots, r)$:*

$$B(x_1, \ldots, x_r) \leq \beta_r(c) \frac{\prod_{k=1}^r x_k^{-1/2+x_k}}{\left(\sum_{k=1}^r x_k\right)^{-1/2+\sum_{k=1}^r x_k}}$$

*with the best possible constant $\beta_r(c) = r^{rc-1/2} c^{(r-1)/2} \frac{(\Gamma(c))^r}{\Gamma(rc)}$.*

It will be convenient to work with Alzer's bound in its log form:

$$\begin{aligned} &\log B(x_1, \ldots, x_r) \\ &\quad \leq \quad \log \beta_r(c) + \left(\sum_{k=1}^r \left(x_k - \frac{1}{2}\right) \log x_k\right) \\ &\quad - \quad \left(-\frac{1}{2} + \sum_{k=1}^r x_k\right) \log \left(\sum_{k=1}^r x_k\right) \end{aligned}$$

with the best possible constant $\log \beta_r(c) = (rc - 1/2) \log r + ((r-1)/2) \log c + r \log \Gamma(c) - \log \Gamma(rc)$.

By choosing $c = \frac{\alpha}{q_i r_i}$ (which is always positive since $\alpha$ must be) this theorem provides an upper bound for $\log B\left(n_{ij1} + \frac{\alpha}{q_i r_i}, \ldots n_{ijr_i} + \frac{\alpha}{q_i r_i}\right)$. With this choice of $c$ we have:

$$\begin{aligned} \log \beta_{r_i}(c) \quad &= \quad (\frac{\alpha}{q_i} - 1/2) \log r_i + ((r_i - 1)/2) \log(\frac{\alpha}{q_i r_i}) \\ &\quad + r_i \log \Gamma(\frac{\alpha}{q_i r_i}) - \log \Gamma(\frac{\alpha}{q_i}) \end{aligned}$$

and so

$$\begin{aligned} &\log B\left(n_{ij1} + \frac{\alpha}{q_i r_i}, \ldots n_{ijr_i} + \frac{\alpha}{q_i r_i}\right) \\ &\quad \leq \quad (\frac{\alpha}{q_i} - 1/2) \log r_i + ((r_i - 1)/2) \log(\frac{\alpha}{q_i r_i}) \\ &\qquad + r_i \log \Gamma(\frac{\alpha}{q_i r_i}) - \log \Gamma(\frac{\alpha}{q_i}) \\ &\qquad + \left(\sum_{k=1}^{r_i} \left(n_{ijk} + \frac{\alpha}{q_i r_i} - \frac{1}{2}\right) \log(n_{ijk} + \frac{\alpha}{q_i r_i})\right) \\ &\qquad - \left(-\frac{1}{2} + \sum_{k=1}^{r_i} n_{ijk} + \frac{\alpha}{q_i r_i}\right) \log \left(\sum_{k=1}^{r_i} n_{ijk} + \frac{\alpha}{q_i r_i}\right) \\ &\quad = \quad (\frac{\alpha}{q_i} - 1/2) \log r_i + ((r_i - 1)/2) \log(\frac{\alpha}{q_i r_i}) + \\ &\qquad r_i \log \Gamma(\frac{\alpha}{q_i r_i}) - \log \Gamma(\frac{\alpha}{q_i}) \\ &\qquad + \left(\sum_{k=1}^{r_i} \left(n_{ijk} + \frac{\alpha}{q_i r_i} - \frac{1}{2}\right) \log(n_{ijk} + \frac{\alpha}{q_i r_i})\right) \\ &\qquad - \left(-\frac{1}{2} + n_{ij} + \frac{\alpha}{q_i}\right) \log \left(n_{ij} + \frac{\alpha}{q_i}\right) \end{aligned} \tag{6}$$

Plugging (6) into (5) and replacing $G$ with $W$ we get:

$$\begin{aligned} &z_i(W) \\ &\quad \leq \quad q^{(+)} \log \Gamma\left(\frac{\alpha}{q_i}\right) - r_i q^{(+)} \log \Gamma\left(\frac{\alpha}{q_i r_i}\right) \\ &\qquad + \sum_{j=1}^{q^{(+)}} \left[(\frac{\alpha}{q_i} - 1/2) \log r_i + ((r_i - 1)/2) \log(\frac{\alpha}{q_i r_i})\right. \\ &\qquad + r_i \log \Gamma(\frac{\alpha}{q_i r_i}) - \log \Gamma(\frac{\alpha}{q_i}) \\ &\qquad + \left(\sum_{k=1}^{r_i} \left(n_{ijk} + \frac{\alpha}{q_i r_i} - \frac{1}{2}\right) \log(n_{ijk} + \frac{\alpha}{q_i r_i})\right) \\ &\qquad \left. - \left(-\frac{1}{2} + n_{ij} + \frac{\alpha}{q_i}\right) \log \left(n_{ij} + \frac{\alpha}{q_i}\right)\right] \\ &\quad = \quad q^{(+)}(\alpha/q_i - 1/2) \log r_i + q^{(+)}(r_i/2 - 1/2) \log(\frac{\alpha}{q_i r_i}) \\ &\qquad + \sum_{j=1}^{q^{(+)}} \left[\left(\sum_{k=1}^{r_i} \left(n_{ijk} + \frac{\alpha}{q_i r_i} - \frac{1}{2}\right) \log(n_{ijk} + \frac{\alpha}{q_i r_i})\right)\right. \\ &\qquad \left. - \left(-\frac{1}{2} + n_{ij} + \frac{\alpha}{q_i}\right) \log \left(n_{ij} + \frac{\alpha}{q_i}\right)\right] \\ &\quad = \quad q^{(+)}(\alpha/q_i - 1/2) \log r_i + q^{(+)}(r_i/2 - 1/2) \log(\frac{\alpha}{q_i r_i}) \\ &\qquad + \sum_{j=1}^{q^{(+)}} \left[\left(\sum_{k=1}^{r_i} \left(n_{ijk} + \frac{\alpha}{q_i r_i}\right) \log(n_{ijk} + \frac{\alpha}{q_i r_i})\right)\right. \end{aligned} \tag{7}$$

$$
\begin{aligned}
&\quad -\left(n_{ij}+\frac{\alpha}{q_i}\right)\log\left(n_{ij}+\frac{\alpha}{q_i}\right)\Bigg] \\
&\quad -\frac{1}{2}\left[\sum_{j=1}^{q^{(+)}}\left(\sum_{k=1}^{r_i}\log(n_{ijk}+\frac{\alpha}{q_i r_i})\right)-\log\left(n_{ij}+\frac{\alpha}{q_i}\right)\right] \\
= \;& q^{(+)}(\alpha/q_i-1/2)\log r_i + q^{(+)}(r_i/2-1/2)\log(\frac{\alpha}{q_i r_i}) \\
&+\sum_{j=1}^{q^{(+)}}\sum_{k=1}^{r_i}\Bigg[\left(n_{ijk}+\frac{\alpha}{q_i r_i}\right)\log(n_{ijk}+\frac{\alpha}{q_i r_i}) \\
&-\left(n_{ijk}+\frac{\alpha}{q_i r_i}\right)\log\left(n_{ij}+\frac{\alpha}{q_i}\right)\Bigg] \\
&-\frac{1}{2}\left[\sum_{j=1}^{q^{(+)}}\left(\sum_{k=1}^{r_i}\log(n_{ijk}+\frac{\alpha}{q_i r_i})\right)-\log\left(n_{ij}+\frac{\alpha}{q_i}\right)\right] \\
= \;& q^{(+)}(\alpha/q_i-1/2)\log r_i + q^{(+)}(r_i/2-1/2)\log(\frac{\alpha}{q_i r_i}) \\
&+\sum_{j=1}^{q^{(+)}}\sum_{k=1}^{r_i}\left(n_{ijk}+\frac{\alpha}{q_i r_i}\right)\log\frac{n_{ijk}+\frac{\alpha}{q_i r_i}}{n_{ij}+\frac{\alpha}{q_i}} \\
&+\frac{1}{2}\left[\sum_{j=1}^{q^{(+)}}\log\left(n_{ij}+\frac{\alpha}{q_i}\right)-\sum_{k=1}^{r_i}\log\left(n_{ijk}+\frac{\alpha}{q_i r_i}\right)\right]
\end{aligned}
\tag{8}
$$

The right-hand side of (8) breaks down naturally into what is almost a prior component [ $q^{(+)}(\alpha/q_i-1/2)\log r_i + q^{(+)}(r_i/2-1/2)\log(\frac{\alpha}{q_i r_i})$] and a strongly data-dependent component (the rest). Note that this 'almost-prior' component is not entirely prior to the data since it depends on the value of $q^{(+)}$, the number of positive values of $n_{ij}$.

To understand the strongly data-dependent component consider the following Bayesian approach to parameter estimation in the saturated model (where no conditional independence relations are assumed). Let $r=\prod_i r_i$ be the number of full joint instantiations of the variables, and associate a Dirichlet parameter $\alpha/r$ with each full joint instantiation. Imagine now updating this Dirichlet prior with the observed data. It is not difficult to see that the posterior mean distribution gives probability $\tilde{p}_\iota = (n_\iota+\alpha/r)/(N+\alpha)$ to each full joint instantiation $\iota$ where $n_\iota$ is the observed frequency of $\iota$ in the data, and $N$ is the size of the data. Marginal probabilities are easy to compute. Let $\tilde{p}_{ijk}$ be the joint probability of variable $i$ taking its $k$th value and its parents (in some fixed graph) taking their $j$th instantiation, then clearly $\tilde{p}_{ijk}=(n_{ijk}+\frac{\alpha}{q_i r_i})/(N+\alpha)$. Similarly, $\tilde{p}_{ij}=(n_{ij}+\frac{\alpha}{q_i})/(N+\alpha)$, where $\tilde{p}_{ij}$ is the probability that the parents take their $j$th instantiation.

From this it is not difficult to see that:

$$
\sum_{j=1}^{q_i}\sum_{k=1}^{r_i}\left(n_{ijk}+\frac{\alpha}{q_i r_i}\right)\log\frac{n_{ijk}+\frac{\alpha}{q_i r_i}}{n_{ij}+\frac{\alpha}{q_i}} = -(N+\alpha)H_{\tilde{p}}(i|W)
$$

where $H_{\tilde{p}}(i|W)$ is the conditional entropy of variable $i$ given its parents $W$. (Note that there already exists valuable work linking BDeu scores to conditional entropy [9].) Since:

$$
\begin{aligned}
&\sum_{j=1}^{q^{(+)}}\sum_{k=1}^{r_i}\left(n_{ijk}+\frac{\alpha}{q_i r_i}\right)\log\frac{n_{ijk}+\frac{\alpha}{q_i r_i}}{n_{ij}+\frac{\alpha}{q_i}} \\
= \;& \sum_{j=1}^{q_i}\sum_{k=1}^{r_i}\left(n_{ijk}+\frac{\alpha}{q_i r_i}\right)\log\frac{n_{ijk}+\frac{\alpha}{q_i r_i}}{n_{ij}+\frac{\alpha}{q_i}} \\
&-\sum_{j=1}^{q^{(0)}}\sum_{k=1}^{r_i}\left(n_{ijk}+\frac{\alpha}{q_i r_i}\right)\log\frac{n_{ijk}+\frac{\alpha}{q_i r_i}}{n_{ij}+\frac{\alpha}{q_i}} \\
= \;& -(N+\alpha)H_{\tilde{p}}(i|W) \\
&-q^{(0)}\frac{\alpha}{q_i}\log(1/r_i)
\end{aligned}
\tag{9}
$$

and $-q^{(0)}\frac{\alpha}{q_i}\log(1/r_i)=\alpha\log r_i-\alpha(q^{(+)}/q_i)\log r_i$, plugging (9) into (8) and rearranging we have:

$$
\begin{aligned}
&z_i(W) \\
&\leq \;\alpha\log r_i + q^{(+)}[(r_i-1)\log(\alpha/q_i r_i)-\log(r_i)]/2 \\
&\quad -(N+\alpha)H_{\tilde{p}}(i|W) \\
&\quad +\frac{1}{2}\left[\sum_{j=1}^{q^{(+)}}\log\left(n_{ij}+\frac{\alpha}{q_i}\right)-\sum_{k=1}^{r_i}\log\left(n_{ijk}+\frac{\alpha}{q_i r_i}\right)\right]
\end{aligned}
\tag{10}
$$

It remains to obtain a easily computable upper bound on the term in square brackets in (10). We have:

$$
\sum_{j=1}^{q^{(+)}}\log\left(n_{ij}+\frac{\alpha}{q_i}\right) = q^{(+)}\log G(n_{ij}+\frac{\alpha}{q_i})_j
$$

where $G(n_{ij}+\frac{\alpha}{q_i})_j$ is the geometric mean of the set $\{n_{ij}+\frac{\alpha}{q_i} : j=1,..q^{(+)}\}$. Since the geometric mean is never greater than the arithmetic mean we have:

$$
\sum_{j=1}^{q^{(+)}}\log\left(n_{ij}+\frac{\alpha}{q_i}\right) \leq q^{(+)}\log(N/q^{(+)}+\alpha/q_i)
$$

Finally a lower bound on $\sum_{j=1}^{q^{(+)}}\sum_{k=1}^{r_i}\log\left(n_{ijk}+\frac{\alpha}{q_i r_i}\right)$ is needed. Suppose $s^{(+)}$ of the $q^{(+)}r_i$ terms are positive, then

$$
\begin{aligned}
&\sum_{j=1}^{q^{(+)}}\sum_{k=1}^{r_i}\log\left(n_{ijk}+\frac{\alpha}{q_i r_i}\right) \\
&\quad = \;(q^{(+)}r_i-s^{(+)})\log\frac{\alpha}{q_i r_i}+\sum_{jk:n_{ijk}\geq 1}\log\left(n_{ijk}+\frac{\alpha}{q_i r_i}\right)
\end{aligned}
$$

where $|\{jk : n_{ijk}\geq 1\}| = s^{(+)}$. The quantity $\sum_{jk:n_{ijk}\geq 1}\log\left(n_{ijk}+\frac{\alpha}{q_i r_i}\right)$ is minimised when the distribution of the $n_{ijk}$ is as 'uneven' as possible, with one of these values having the value $N-s^{(+)}$ and all others with the value 1. So we have:

$$
\begin{aligned}
&\sum_{jk:n_{ijk}\geq 1}\log\left(n_{ijk}+\frac{\alpha}{q_i r_i}\right) \\
&\quad \geq \;(s^{(+)}-1)\log\left(1+\frac{\alpha}{q_i r_i}\right)+\log\left(N-s^{(+)}+\frac{\alpha}{q_i r_i}\right)
\end{aligned}
$$

and therefore:

$$
-\sum_{j=1}^{q^{(+)}}\sum_{k=1}^{r_i}\log\left(n_{ijk}+\frac{\alpha}{q_i r_i}\right)
$$

$$
\begin{aligned}
&\leq -(q^{(+)}r_i - s^{(+)}) \log \frac{\alpha}{q_i r_i} \\
&\quad -(s^{(+)} - 1) \log \left(1 + \frac{\alpha}{q_i r_i}\right) \\
&\quad - \log \left(N - s^{(+)} + \frac{\alpha}{q_i r_i}\right)
\end{aligned} \tag{11}
$$

Applying (11) gives the following upper bound on $z_i(W)$:

$$
\begin{aligned}
&z_i(W) \\
&\leq \alpha \log r_i + q^{(+)}[(r_i - 1)\log(\alpha/q_i r_i) - \log r_i]/2 \\
&\quad -(N + \alpha) H_{\tilde{p}}(i|W) \\
&\quad +\frac{1}{2} q^{(+)} \log(N/q^{(+)} + \alpha/q_i) \\
&\quad -\frac{1}{2}(q^{(+)}r_i - s^{(+)}) \log \frac{\alpha}{q_i r_i} \\
&\quad -\frac{1}{2}(s^{(+)} - 1) \log \left(1 + \frac{\alpha}{q_i r_i}\right) \\
&\quad -\frac{1}{2} \log \left(N - s^{(+)} + \frac{\alpha}{q_i r_i}\right)
\end{aligned}
$$

which can be rearranged to give:

$$
\begin{aligned}
&2z_i(W) \\
&\leq 2[\alpha \log r_i - (N + \alpha) H_{\tilde{p}}(i|W)] \\
&\quad + \left(s^{(+)} - q^{(+)}\right) \log \left(\frac{\alpha}{q_i r_i}\right) \\
&\quad +q^{(+)} \log \left(\frac{N}{q^{(+)} r_i} + \frac{\alpha}{q_i r_i}\right) \\
&\quad - \left(s^{(+)} - 1\right) \log \left(1 + \frac{\alpha}{q_i r_i}\right) \\
&\quad - \log \left(N - s^{(+)} + \frac{\alpha}{q_i r_i}\right)
\end{aligned} \tag{12}
$$

From (12) it is clear that $\frac{\alpha}{q_i r_i}$ is a key quantity. Since $q_i$ grows exponentially with the number of parents, $\frac{\alpha}{q_i r_i}$ will be very small for large parent sets and in such cases $\log\left(\frac{\alpha}{q_i r_i}\right)$ will be a highly negative number. Consider now the term $\left(s^{(+)} - q^{(+)}\right) \log\left(\frac{\alpha}{q_i r_i}\right)$. We have that $s^{(+)} - q^{(+)} \geq 0$ since for each positive $n_{ij}$ there must be at least one positive $n_{ijk}$. We have $s^{(+)} - q^{(+)} = 0$ only in the extreme case where, for each parent instantiation for which we have observed data ($n_{ij} > 0$), all the associated datapoints have the same value for the child $i$. In other words, in the observed data, the joint value of the parents *determines* that of the child. In this unusual case even a very negative value of $\log\left(\frac{\alpha}{q_i r_i}\right)$ does not drive the upper bound down. When the data rule out a deterministic relation between parents and child we have $s^{(+)} - q^{(+)} > 0$ and $\log\left(\frac{\alpha}{q_i r_i}\right)$ will push down the upper bound. In summary, the term $\left(s^{(+)} - q^{(+)}\right) \log\left(\frac{\alpha}{q_i r_i}\right)$ penalises parent sets according to how much the values of the parents fail to determine that of the child.

Further insight into this issue can be gained by rewriting (7) as:

$$
\begin{aligned}
&z_i(W) \\
&\leq q^{(+)}(\alpha/q_i - 1/2) \log r_i + q^{(+)}(r_i/2 - 1/2) \log \left(\frac{\alpha}{q_i r_i}\right) \\
&\quad + \sum_{j=1}^{q^{(+)}} \sum_{k=1}^{r_i} \left[ \left(n_{ijk} + \frac{1}{r_i}\left(\frac{\alpha}{q_i} - \frac{1}{2}\right)\right) \log \left(\frac{n_{ijk} + \frac{\alpha}{q_i r_i}}{n_{ij} + \frac{\alpha}{q_i}}\right) \right. \\
&\quad \left. -\frac{1}{2}\left(1 - \frac{1}{r_i}\right) \log \left(n_{ijk} + \frac{\alpha}{q_i r_i}\right) \right]
\end{aligned} \tag{13}
$$

For each $n_{ijk}$, if $n_{ijk} > 0$ then $n_{ijk} \geq 1 > ((1/2) - (\alpha/q_i))/r_i$ and both terms of the summand for $n_{ijk}$ are negative, driving down the upper bound. (So if all $n_{ijk}$ are positive a simple upper bound on $z_i(W)$ is obtained by deleting the double summation in (13)). However for each $n_{ijk}$ where $n_{ijk} = 0, n_{ij} > 0$ the second term is positive if $\alpha/q_i < r_i$ and the first term also if $\alpha/q_i < 1/2$. Both of these conditions will hold for typical choices of $\alpha$ ($\alpha = 1$, for example) and the upper bound will be pushed up. Each $n_{ijk}$ for which $n_{ijk} = 0, n_{ij} > 0$ is an example of 'determinism in the data': the $k$th child value never occurs when the parents are in their $j$th configuration (and this $j$th configuration occurs at least once in the data). The parents are 'rewarded' each time this occurs by a positive boost to the upper bound on their score.

## 4 How tight are the bounds?

In this section local scores are compared against their upper bounds as computed by the direct application of Azler's bound (8) and also using the less data-dependent bound given in (12). A selection of such comparisons is presented here with the aim of exemplifying the main points.

In one experiment 100 datapoints were sampled from the well-known 'Asia' network and local scores for all possible parent sets were computed for the variable *Dyspnea* with $\alpha$ set to 1. Fig 1 shows the 128 local scores ordered by their value together with the upper bound computed by (8) and by (12), labelled `true`, `alzer` and `easy` respectively. Figs 2–8 shows results in the same form, for different numbers of datapoints, values of $\alpha$ and Bayesian network. See figure captions for details. The most strking finding is that both bounds become much tighter as the amount of data increases. This is as expected, since from (12) we can see that the entropy dominates as $N$ increases. Such asymptotic behaviour has been analysed in some detail by Ueno [9].

## 5 Potential for pruning the search for local scores

A key motivation for obtaining bounds on BDeu local scores is to be able to (cheaply) prune the search for optimal parent sets. de Campos and Ji [6] have already achieved impressive pruning results.

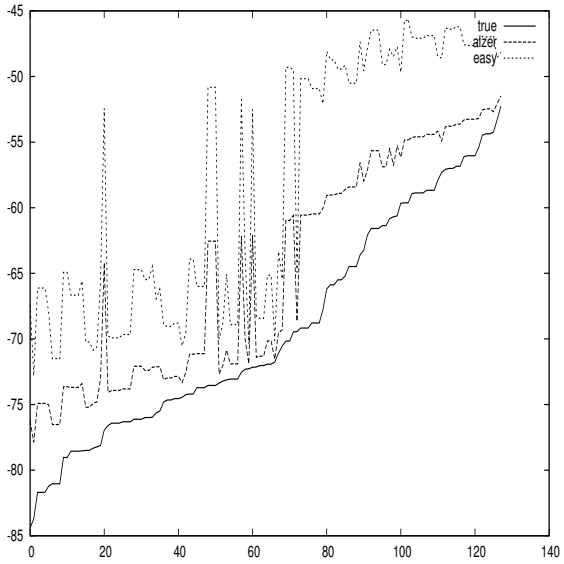



**Figure 1.** Local scores and upper bounds for parent sets of the variable *Dyspnea* in the 'Asia' network. Using 100 datapoints sampled from the 'Asia' network and $\alpha = 1$

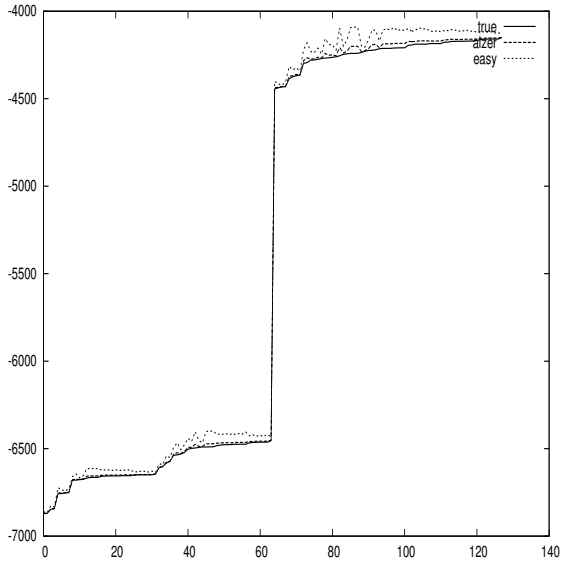



**Figure 2.** Local scores and upper bounds for parent sets of the variable *Dyspnea* in the 'Asia' network. Using 10000 datapoints sampled from the 'Asia' network and $\alpha = 1$

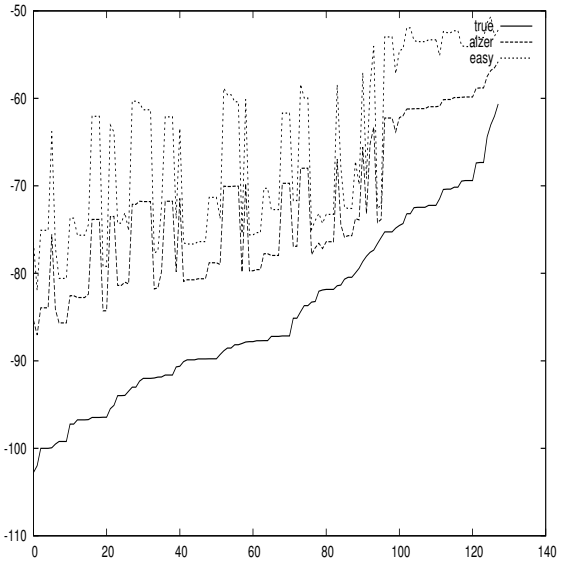



**Figure 3.** Local scores and upper bounds for parent sets of the variable *Dyspnea* in the 'Asia' network. Using 100 datapoints sampled from the 'Asia' network and $\alpha = 0.01$

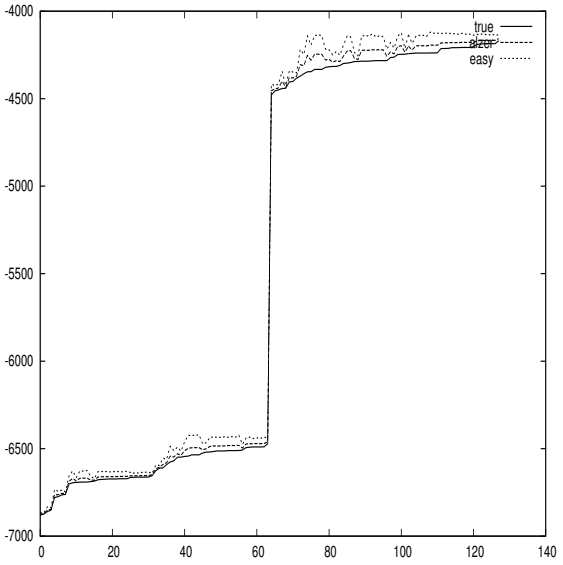



**Figure 4.** Local scores and upper bounds for parent sets of the variable *Dyspnea* in the 'Asia' network. Using 10000 datapoints sampled from the 'Asia' network and $\alpha = 0.01$

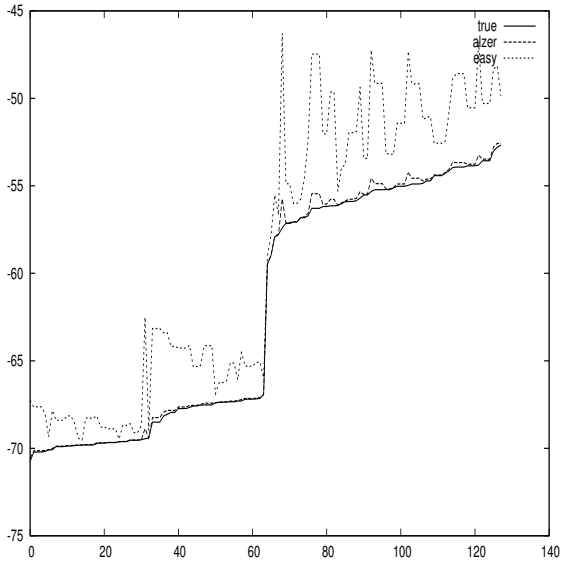



**Figure 5.** Local scores and upper bounds for parent sets of the variable *Dyspnea* in the 'Asia' network. Using 100 datapoints sampled from the 'Asia' network and $\alpha = 100$

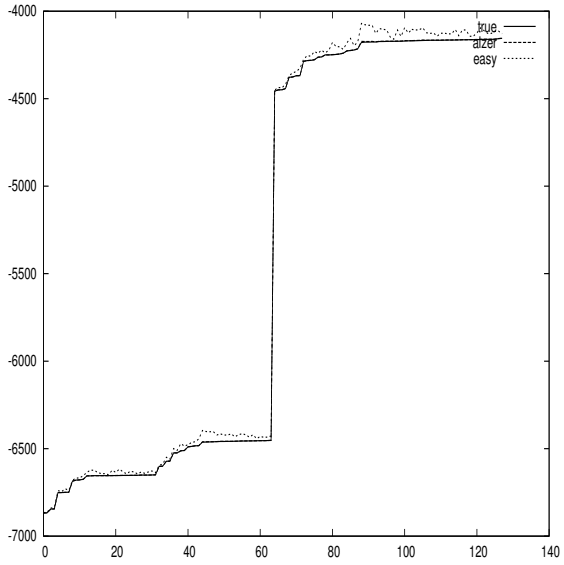



**Figure 6.** Local scores and upper bounds for parent sets of the variable *Dyspnea* in the 'Asia' network. Using 10000 datapoints sampled from the 'Asia' network and $\alpha = 100$

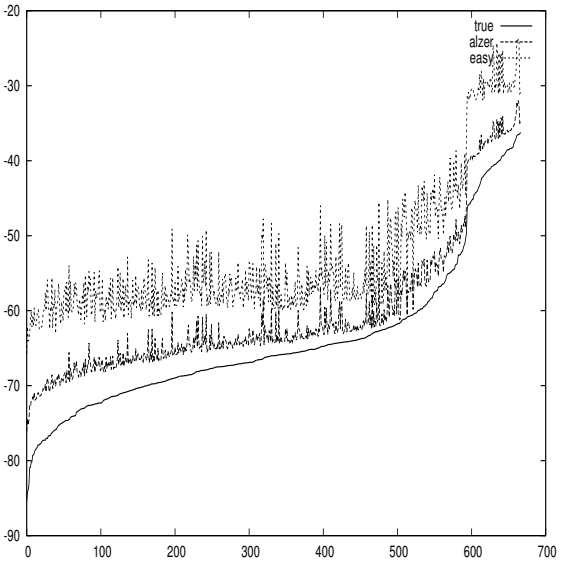



**Figure 7.** Local scores and upper bounds for small parent sets of the variable *HYPOVOLEMIA* in the 'alarm' network. Using 100 datapoints sampled from the 'alarm' network and $\alpha = 1$

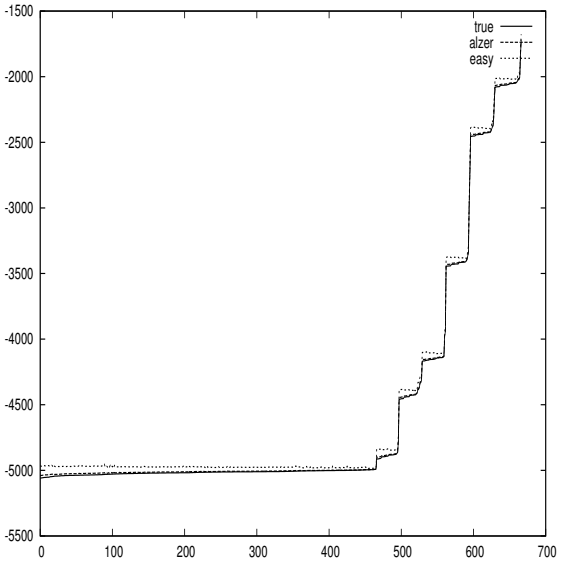



**Figure 8.** Local scores and upper bounds for small parent sets of the variable *HYPOVOLEMIA* in the 'alarm' network. Using 10000 datapoints sampled from the 'alarm' network and $\alpha = 1$

The bound (8), equivalent to (10), is tight but not appropriate for pruning. The looser bound (12) depends only on: $q^{(+)}(W), s^{(+)}(W), q_i$ and $H_{\tilde{p}}(i|W)$. These quantities can be bounded by the corresponding quantities for subsets and supersets. Let $L$ and $U$ be such that $L \subset W \subset U$. Abbreviate $(\min_{i' \in W \setminus L} r_{i'}) q_i(L)$ to $q'_i(L)$ and $(\min_{i' \in U \setminus W} r_{i'})^{-1} q_i(U)$ to $q'_i(U)$. We have

- $q'_i(L) \leq q_i(W) \leq q'_i(U)$
- $q^{(+)}(L) \leq q^{(+)}(W) \leq q^{(+)}(U)$
- $s^{(+)}(L) \leq s^{(+)}(W) \leq s^{(+)}(U)$
- $s^{(+)}(L) - q^{(+)}(L) \leq s^{(+)}(W) - q^{(+)}(W) \leq s^{(+)}(U) - q^{(+)}(U)$
- $-H_{\tilde{p}}(i|L) \leq -H_{\tilde{p}}(i|W) \leq -H_{\tilde{p}}(i|U)$

Note that the last double inequality is the well-known result that conditional entropy is always non-increasing as variables are added to the conditioning set. Conditional entropy only remains constant if the relevant conditional independence relation obtains.

If $\frac{\alpha}{q_i r_i} \leq 1$ there is the following bound:

$$
\begin{aligned}
& 2z_i(W) \\
& \quad \leq \quad 2[\alpha \log r_i - (N+\alpha) H_{\tilde{p}}(i|U)] \\
& \qquad + \left(s^{(+)}(L) - q^{(+)}(L)\right) \log \left( \frac{\alpha}{q'_i(L) r_i} \right) \\
& \qquad + q^{(+)}(U) \log \left( \frac{N}{q^{(+)}(L) r_i} + \frac{\alpha}{q'_i(L) r_i} \right) \\
& \qquad - \left(s^{(+)}(L) - 1\right) \log \left( 1 + \frac{\alpha}{q'_i(U) r_i} \right) \\
& \qquad - \log \left( N - s^{(+)}(U) + \frac{\alpha}{q'_i(U) r_i} \right)
\end{aligned}
\tag{14}
$$

If the bound for $z_i(W)$ given by (14) is less than $z_i(L)$ it follows that $W$ cannot be an optimal parent set for variable $i$. An obvious choice for $U$ is $\{1, \ldots, p\} \setminus \{i\}$. The next step in this work is to see to what extent (14) provides effective pruning and compare to the existing method of de Campos and Ji [6].

## ACKNOWLEDGEMENTS

I would like to thank the referees for their comments, which helped improve this paper considerably. This work has been supported by the UK Medical Research Council (Project Grant G1002312).

## REFERENCES

[1] Horst Alzer. Inequalities for the Beta function of $n$ variables. *The ANZIAM Journal*, 44:609–623, 2003.

[2] Gregory F. Cooper and Edward Herskovits. A Bayesian method for the induction of probabilistic networks from data. *Machine Learning*, 9:309–347, 1992.

[3] James Cussens. Bayesian network learning by compiling to weighted MAX-SAT. In *Proceedings of the 24th Conference on Uncertainty in Artificial Intelligence (UAI 2008)*, pages 105–112, Helsinki, 2008. AUAI Press.

[4] James Cussens. Bayesian network learning with cutting planes. In Fabio G. Cozman and Avi Pfeffer, editors, *Proceedings of the 27th Conference on Uncertainty in Artificial Intelligence (UAI 2011)*, pages 153–160, Barcelona, 2011. AUAI Press.

[5] C de Campos, Z Zeng, and Q Ji. Structure learning of Bayesian networks using constraints. In *Proc. of the 26th International Conference on Machine Learning*, 2009.

[6] Cassio P. de Campos and Qiang Ji. Properties of Bayesian Dirichlet scores to learn Bayesian network structures. In *Proc. AAAI-10*, pages 431–436, 2010.

[7] David Heckerman, Dan Geiger, and David M. Chickering. Learning Bayesian networks: The combination of knowledge and statistical data. *Machine Learning*, 20(3):197–243, 1995.

[8] M. Teyssier and D. Koller. Ordering-based search: A simple and effective algorithm for learning Bayesian networks. In *Proc. UAI 2005*, pages 584–590, 2005.

[9] Maomi Ueno. Learning networks determined by the ratio of prior and data. In *Proc. UAI-2010*, pages 598–605, 2010.

# New Local Move Operators for Learning the Structure of Bayesian Networks

**Jimmy Vandel** and **Brigitte Mangin** and **Simon de Givry**
**UBIA, UR 875, INRA, F-31320 Castanet Tolosan, France**
**{jvandel,mangin,degivry}@toulouse.inra.fr**

**Abstract.** We propose new local move operators incorporated into a score-based stochastic greedy search algorithm to efficiently escape from local optima in the search space of directed acyclic graphs. We extend the classical set of arc addition, arc deletion, and arc reversal operators with a new operator replacing or *swapping* one parent to another for a given node, *i.e.* combining two elementary operations (arc addition and arc deletion) in one move. The old and new operators are further extended by doing more operations in a move in order to overcome the acyclicity constraint of Bayesian networks. These extra operations are temporally performed in the space of directed *cyclic* graphs. At the end acyclicity is restored and newly defined operators actually lead to a gain in graph score. Our experimental results on standard Bayesian networks and challenging gene regulatory networks show large BDeu score and recall value improvements compared to state-of-the-art structure learning algorithms when the sample size is small.

## 1 Introduction

Learning the structure of Bayesian networks from fully observed data is known to be an NP-hard problem [5] which has received a lot of attention from researchers during the last two decades [8]. Due to its difficulty, heuristic methods have been widely used to learn Bayesian network structures. Two main approaches have been studied: constraint-based methods and score-based methods [18]. Constraint-based methods aim at satisfying as much independence present in the data as possible using conditional independence tests. Unfortunately, these methods can be sensitive to failures in individual independence tests. Score-based methods use a scoring function $f$ to score a network structure with respect to the data. They score the whole network at once, being therefore less sensitive to individual failures. Different Bayesian and non-Bayesian scoring metrics can be used such as the *Bayesian Information Criterion* (BIC) [26] or the *Bayesian Dirichlet criterion* (BDeu) [16]. Score-based methods explore the space of network structures to find the highest-scoring network. This space being superexponential in the number of variables, local search methods are used such as greedy ascent search (also called hill-climbing), tabu search, simulated-annealing, and other complex metaheuristics like ant colony optimization [7]. In spite of its simplicity, the (repeated randomized or *stochastic*) greedy search method reveals to be a competitive method compared to more complex algorithms [12]. Starting from an initial network structure, it performs a series of local moves until a local optimum is found. Each move selects and applies the best elementary operation(s) on the current network structure. The set of candidate neighboring structures is called the *neighborhood* in the sequel. A classical neighborhood is composed of single arc additions, deletions, and reversals. Using *larger* neighborhoods efficiently allows to find better local optima, and thus better network structures.

[20] proposed an optimal reinsertion of a target node by removing all its edges and reinserting it optimally. However this approach is limited to small problems only. [17] used a restricted form of look ahead called LAGD, combining several operations in a single move. In this paper, we follow a similar approach by focusing our local operations on a target node and combining several operations guided by the global acyclicity constraint of Bayesian networks. By doing so, we are able to exploit large neighborhoods efficiently. Other approaches use a compact representation of a set of network structures. They explore either the search space of variable orderings (an optimal structure compatible with the order being easier to find) [6, 28, 25, 2], or the search space of Markov-equivalent network classes like *Greedy Equivalence Search* (GES) [3, 21, 7].

In Section 2, we give Bayesian network background. Next, we define a specific stochastic greedy search algorithm and introduce the new local move operators in Section 3. We report experimental results in Section 4 and conclude.

## 2 Bayesian network structure learning

A Bayesian network [18] denoted by $B = (G, \mathbf{P}_G)$ is composed of a directed acyclic graph (DAG) $G = (\mathbf{X}, \mathbf{E})$ with nodes representing $p$ random discrete variables $\mathbf{X} = \{X_1, \ldots, X_p\}$, linked by a set of directed edges or *arcs* $\mathbf{E}$[1], and a set of conditional probability distributions $\mathbf{P}_G = \{P_1, \ldots, P_p\}$ defined by the topology of the graph: $P_i = \mathbb{P}(X_i|Pa(X_i))$ where $Pa(X_i) = \{X_j \in \mathbf{X} \,|\, (X_j \rightarrow X_i) \in \mathbf{E}\}$ is the set of parent nodes of $X_i$ in $G$. A Bayesian network $B$ represents a joint probability distribution on $\mathbf{X}$ such that: $\mathbb{P}(\mathbf{X}) = \prod_{i=1}^{p} \mathbb{P}(X_i|Pa(X_i))$.

Conditional probability distributions $\mathbf{P}_G$ are determined by a set of parameters. Given the network structure $G$, and the fully observed data $D$, parameters can be estimated by simple counting, following the maximum likelihood principle.

Learning the structure of a Bayesian network consists in finding a DAG $G$ maximizing the posterior distribution $\mathbb{P}(G|D)$. We have $\mathbb{P}(G|D) \propto \mathbb{P}(D|G)\mathbb{P}(G)$ since $\mathbb{P}(D)$ is independent of $G$. Under specific assumptions, the *marginal loglikelihood* $\log(\mathbb{P}(D|G))$ can be expressed as a consistent decomposable scoring function $f$, such as the BIC and BDeu criteria [3] :

[1] In the paper, we use $G = \mathbf{E}$ when the set of nodes is implicit.

$$f(D,G) = \sum_{i=1}^{p} f_{X_i}(D,G) = \sum_{i=1}^{p} f_{X_i}(D, Pa(X_i)) \quad (1)$$

As $f$ is consistent, maximizing $f$ when the sample size tends to infinity leads to select the true structure or one among the Markov-equivalent structures set. A set of Bayesian networks are Markov-equivalent if they imply exactly the same set or *map* of independence constraints among variables[2]. Next, we describe a novel greedy search method maximizing $f$ in the space of DAGs.

## 3 Stochastic Greedy Search

We define the Stochastic Greedy Search (SGS) algorithm[3] for structural learning of Bayesian networks in Algorithm 1. It collects the best DAG found by $r$ randomized hill climbing algorithms. Stochasticity comes from two random draws. The first one, common in the structure learning community, is due to `InitGraph` that returns a random DAG used by the inner greedy search loop. The second is more original. It is the random DAG drawn among the best neighbors in the neighborhood of the current DAG $G$ ( `SelectRandom` at line 1 of Algorithm 1). The neighborhood of $G$, returned by `Neighbors`, is composed of the usual elementary operations on DAGs: arc addition (ADD), arc deletion (DELETE), and arc reversal (REVERSE). This classical neighborhood is denoted $\mathcal{N}^{ADR}$ in the sequel. Only feasible operations are considered, which do not create cycles. In the next subsections, we are going to extend this set of operations. We empirically observe that the first random draw may be counter-productive and starting from an empty graph is often better than from a random DAG, as also observed in [19].

**Algorithm 1**: Stochastic Greedy Search algorithm.

```
Input  : An iid sample D, a scoring function f, number of
         repeats of different GS r
Output : A directed acyclic graph
G* ← ∅ /* Best DAG initialized with the empty graph */;
s* ← f(D, G*) /* score of the empty graph */;
/* Repeat r randomized greedy searches */;
for i ← 1 to r do
    G ← InitGraph() /* Choose an initial DAG */;
    s ← f(D, G) ;
    repeat
        improvement ← false ;
        s^max ← max_{G' ∈ Neighbors(G)} f(D, G');
        if s^max > s then
            /* Select at random among the best neighbors */;
            G^max ← {G' ∈ Neighbors(G) | f(D, G') = s^max} ;
1           G ← SelectRandom(G^max) ;
            s ← s^max ;
            improvement ← true ;
    until ¬ improvement ;
    /* Keep the best DAG of r greedy searches */;
2   if s > s* then
        G* ← G ;
        s* ← s ;
return G* ;
```

**Proposition 1.** *Let $D$ be a dataset of $n$ records that are identically and independently sampled from some distribution $\mathbb{P}(\cdot)$. Let $f$ be a locally consistent scoring function. The inner loop of the SGS algorithm returns a minimal independence map of $\mathbb{P}(\cdot)$ as the sample size $n$ grows large.*

The local consistency of $f$ ensures that adding any arc that eliminates an independence constraint that does not hold in the generative distribution $\mathbb{P}(\cdot)$ increases the score. Conversely, deleting any arc that results in a new independence constraint that holds in $\mathbb{P}(\cdot)$ also increases the score. Hence, starting from any DAG, by selecting strictly improving moves, the algorithm terminates (because of a finite number of DAGs) and returns an independence map of $\mathbb{P}(\cdot)$ (*I-map of* $\mathbb{P}(\cdot)$: every independence implied by the resulting DAG is verified in $\mathbb{P}(\cdot)$, in the worst case it can be a complete graph) which is minimal thanks to arc deletion operations and local consistency.

The main interest of our randomization approach is to simulate a search in the space of score-equivalent networks. Each greedy search moves from a randomly-selected DAG instance of a Markov-equivalence class $\mathcal{E}(G)$ to another randomly-selected DAG of an *adjacent*[4] Markov-equivalence class $\mathcal{E}(G')$ thanks to our `SelectRandom` function. It results in a stronger property:

**Proposition 2.** *Let $D$ be a dataset of $n$* iid *fully observed samples of some* faithful *distribution $\mathbb{P}(\cdot)$. Let $f$ be a locally consistent scoring function. SGS returns a perfect map of $\mathbb{P}(\cdot)$ as both the sample size $n$ and the number of restarts $r$ grow large.*

Recall that a faithful distribution admits a unique perfect map corresponding to the optimal structure. Compared to the GES algorithm [3], which offers the same optimality guarantee within a two-phase greedy search, SGS chooses the orientation of some *compelled arcs*[5] of the *true* DAG at random, whereas GES waits while no *v-structures* impose orientation constraints. See an example in Figure 1.

Notice that neither GES nor SGS find an optimal structure in polynomial time in the worst case, even when using a constant time consistent scoring function and a faithful distribution[6]. In general, without the faithfulness assumption, learning the optimal structure is NP-hard even when the sample size is large and when each node has at most $k$ parents, for all $k \geq 3$ [4].

We observe in the experiments that a small number of restarts $r$ allows to find DAGs with better scores than GES, especially when the sample size $n$ is limited, in this case GES found a local optimum and SGS is able to find other better local optima thanks to randomization. This was also observed in [21].

When the sample size is small the learning problem becomes more difficult: the empirical distribution may be far from a perfect map resulting in many local optima and the scoring function is no more consistent, *i.e.* the likelihood does not dominate the penalty term of the scoring function which increases with the parent variable domain sizes [3]. In this complex situation, we propose a new operator to escape from some local optima.

### 3.1 SWAP operator

Consider the 3-variable example in Figure 2 with observed data $D$, scoring function $f$, and initial DAG $G_0 = \{X2 \rightarrow X3\}$. Let assume $f(D, \{X1 \rightarrow X3\}) > f(D, \{X2 \rightarrow X3\}) > f(D, \{X1 \rightarrow X3, X2 \rightarrow$

[2] BIC/BDeu give the same score for Markov-equivalent DAGs.

[3] Don't mistake for the SGS algorithm in [27] which is a constraint-based algorithm, but ours is score-based.

[4] Two equivalence classes $\mathcal{E}(G)$, $\mathcal{E}(G')$ are adjacent iff $G$ is an I-map of $G'$ or vice-versa and the number of edges in the graphs $G$ and $G'$ differs by one.

[5] An arc $X \rightarrow Y$ in $G$ is *compelled* if that arc exists in every DAG of $\mathcal{E}(G)$, otherwise it is said *reversible*.

[6] The number of possible sets $S$ in the GES operator ADD($E$,$S$) is exponential in the maximum degree $d$ of the current graph in the worst case and $r$ is unbounded for SGS.

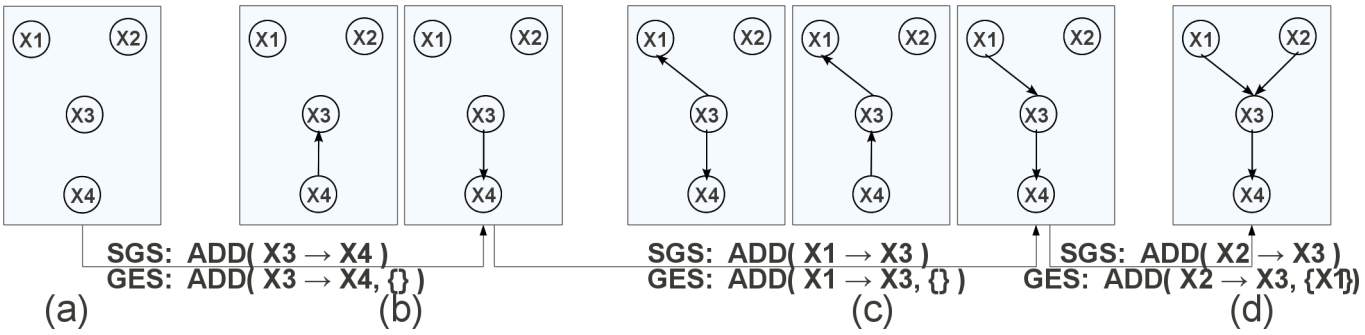



**Figure 1.** Four adjacent Markov-equivalence classes found by GES during its first phase of edge and v-structure insertions. (a) GES and SGS start from the empty graph . (d) The true DAG is found after three moves. The orientation of $X3 \rightarrow X4$ and $X1 \rightarrow X3$ edges are chosen at random by SGS, whereas GES waits until its third move to decide on edge orientations based on DAG score comparisons (enforcing the v-structure $X1 \rightarrow X3 \leftarrow X2$ as stated by the extra ADD parameter $\{X1\}$, and forbidding $X1 \rightarrow X3 \leftarrow X4$ in its second move).

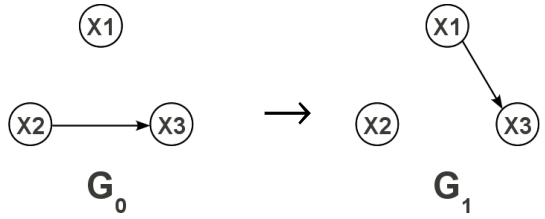



**Figure 2.** The operator SWAP($X2|X1 \rightarrow X3$) applied to a 3-variable problem.

$X3\}) > f(D, \{X3 \rightarrow X1, X2 \rightarrow X3\}) > f(D, \{X2 \rightarrow X1, X2 \rightarrow X3\}) > f(D, \emptyset)$. Then $G_0$ is a local minimum for the classical neighborhood $\mathcal{N}^{ADR}$. Our new operator, denoted SWAP($X|Y \rightarrow Z$), consists in changing one parent $X$ to another parent $Y$ for one target node $Z$. This is equivalent to a simultaneous pair of ADD and DELETE operators restricted to the same target node. In our example, applying SWAP($X2|X1 \rightarrow X3$) corresponds to DELETE($X2 \rightarrow X3$),ADD($X1 \rightarrow X3$), resulting in the better DAG $G_1 = \{X1 \rightarrow X3\}$ as shown in Figure 2. The extended neighborhood using the four operators is denoted $\mathcal{N}^{ADRS}$ and SGS using $\mathcal{N}^{ADRS}$ (respectively $\mathcal{N}^{ADR}$) is denoted SGS$^2$ (Stochastic Greedy Search with Swap) (resp. SGS$^1$) in the sequel.

A typical suboptimality problem that we observed in our experiments happens when two nodes have the same parents. Figure 3 shows such an example with four variables. Because child nodes $X3$ and $X4$ are highly correlated due to their common parents $X1$ and $X2$, the first arc usually added is either $X3 \rightarrow X4$ or $X4 \rightarrow X3$ especially if the conditional probability distributions of $X3$ and $X4$ given $X1$ and $X2$ are close. Then adding a first parent to $X3$ and furthermore a second parent to $X4$ and $X3$ give the DAG $G_4$. Here deleting $X3 \rightarrow X4$ is going to decrease the score and the same negative effect results when adding $X1 \rightarrow X4$ because it leads to a three-parent node. For this node, the increase in likelihood may not overcome the penalization term. But doing both operations simultaneously thanks to our SWAP operator results in a better local optimum DAG.

Let $p$ be the number of variables in the current DAG and $k$ be the maximum number of parents per node. Assuming a sparse graph, $p \gg k$, the number of SWAP operations is bounded by $O(kp^2)$ in the extended neighborhood $\mathcal{N}^{ADRS}$, whereas it is bounded by $O(p^2)$ for ADD and $O(kp)$ for DELETE and REVERSE. The complexity of $\mathcal{N}^{ADRS}$ is therefore in $O(kp^2)$, whereas it is in $O(p^2)$ for the classical neighborhood $\mathcal{N}^{ADR}$. Notice that other approaches using larger neighborhoods such as *h-look ahead in l good directions* (LAGD) has a worst-case complexity in $O(l^{h-1}p^2)$ [17] and optimal reinsertion (OR) neighborhood is in $O(2^k p^{k+1})$ [20]. In particular, LAGD complexity does not benefit from sparsity, $l^{h-1}$ being constant, whereas our approach is faster when $k$ decreases. Moreover computing the score difference of two DAGs before and after a SWAP operation is easy to do as it remains local to a single target node thanks to score decomposition.

Another source of suboptimality comes from the global acyclicity constraint of Bayesian networks.

## 3.2 Breaking cycles by successive deletions and swaps

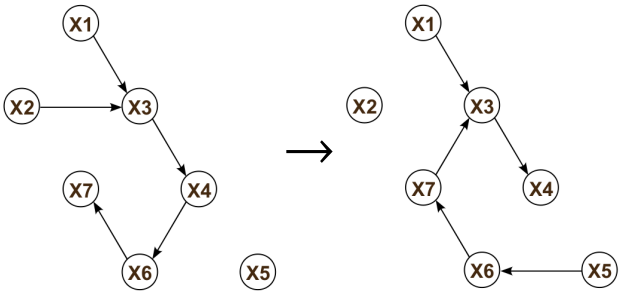



**Figure 4.** Applying an extended SWAP* operation breaking a cycle by an additional SWAP operation:
SWAP*($X2|X7 \rightarrow X3$) = {SWAP($X2|X7 \rightarrow X3$), SWAP($X4|X5 \rightarrow X6$)}.

Consider the 7-variable DAG example in Figure 4. Swapping the parent $X2$ of $X3$ by $X7$ in DAG $G$ (Fig. 4.left) introduces a directed cycle $\{X7 \rightarrow X3, X3 \rightarrow X4, X4 \rightarrow X6, X6 \rightarrow X7\}$ and is therefore forbidden in our $\mathcal{N}^{ADRS}$ neighborhood. However it may correspond to a large *local* score improvement with respect to variable $X3$. Let us denote this improvement by $\Delta_{X3}(G, \text{SWAP}(X2|X7 \rightarrow X3)) = f_{X3}(D, G') - f_{X3}(D, G)$ with $G'$ obtained by applying the SWAP operation on $G$ ($G'$ is not a valid DAG), and $D$ and $f$ being the sample and scoring function. Our idea is to heuristically guide the search for a second (or more) local operator to be applied on $G'$ in order

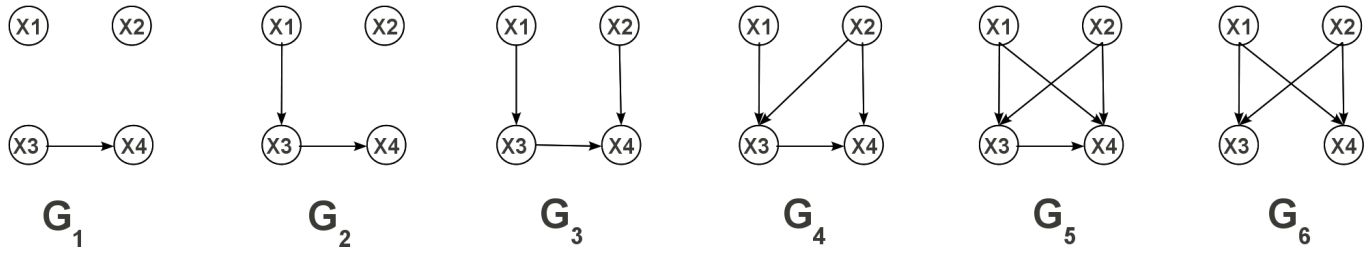



**Figure 3.** Problems with the classical neighborhood $\mathcal{N}^{ADR}$ when two nodes (here $X3$ and $X4$) have the same parents. True DAG is $G_6$. Starting from the empty graph (not shown), $G_4$ can be locally optimum for $\mathcal{N}^{ADR}$ when the sample size is small (score of $G_5$ lower than $G_4$), whereas in $\mathcal{N}^{ADRS}$, SWAP($X3|X1 \rightarrow X4$) moves directly from $G_4$ to $G_6$.

to restore graph acyclicity ($G'$ becomes valid) and such that the *true* score of the final DAG is greater than the score of the original one. In Figure 4, it is obtained by applying a second SWAP.

**Algorithm 2**: SWAP*($X|Y \rightarrow Z$) operator.

**Input** : operation $X|Y \rightarrow Z$, sample $D$, score $f$, DAG $G(\mathbf{X}, \mathbf{E})$
**Output**: a set of local operations $\mathbf{L}$
$\mathbf{L} \leftarrow \emptyset$ /* *Initialize output operations to the empty set* */ ;
$\mathbf{X'} \leftarrow \mathbf{X}$ /* *Candidate parent set for future swaps* */ ;
$G' \leftarrow G$ /* *Copy of input DAG* */ ;
3 $\Delta = \Delta_Z(G', \text{SWAP}(X|Y \rightarrow Z))$ /* *Putative score improvement* */ ;
**if** $\Delta > 0$ **then**
  $\mathbf{L} \leftarrow \mathbf{L} \cup \{\text{SWAP}(X|Y \rightarrow Z)\}$ ;
  Apply SWAP($X|Y \rightarrow Z$) to $G'$ ;
  /* *Repeat deletion or swap operations until no more cycles*
4  */ **while** $\Delta > 0 \wedge (\boldsymbol{C} \leftarrow \texttt{NextCycle}(G')) \neq \emptyset$ **do**
5  $\mathbf{X'} \leftarrow \mathbf{X'} \setminus nodes(\mathbf{C})$ ;
  /* *Choose the best deletion to break cycle* $\boldsymbol{C}$ */ ;
6  $(U^* \rightarrow W^*) \leftarrow$
  $\text{argmax}_{(U \rightarrow W) \in \mathbf{C} \setminus \{Y \rightarrow Z\}} \Delta_W(G', \text{DELETE}(U \rightarrow W))$ ;
  /* *Test if the sum of local score changes is positive* */ ;
  **if** $\Delta + \Delta_{W^*}(G', DELETE(U^* \rightarrow W^*)) > 0$ **then**
    $\mathbf{L} \leftarrow \mathbf{L} \cup \{\text{DELETE}(U^* \rightarrow W^*)\}$ ;
    $\Delta \leftarrow \Delta + \Delta_{W^*}(G', \text{DELETE}(U^* \rightarrow W^*))$ ;
    Apply DELETE($U^* \rightarrow W^*$) to $G'$ ;
  **else**
    /* *Choose the best swap to get a positive change* */ ;
7  $(U^*|V^* \rightarrow W^*) \leftarrow$
    $\text{argmax}_{(U \rightarrow W) \in \mathbf{C}, V \in \mathbf{X'}} \Delta_W(G', \text{SWAP}(U|V \rightarrow W))$ ;
    $\Delta \leftarrow \Delta + \Delta_{W^*}(G', \text{SWAP}(U^*|V^* \rightarrow W^*))$ ;
    **if** $\Delta > 0$ **then**
      $\mathbf{L} \leftarrow \mathbf{L} \cup \{\text{SWAP}(U^*|V^* \rightarrow W^*)\}$ ;
      Apply SWAP($U^*|V^* \rightarrow W^*$) to $G'$ ;
    **else**
8  $\mathbf{L} \leftarrow \emptyset$ /* *Abort all local operations* */ ;
**return** $\mathbf{L}$ ;

For that purpose, we define an extended SWAP operator, denoted SWAP*, able to break all directed cycles by performing a succession of deletion or swap operations. It can be seen as a kind of greedy descent search in the space of directed cyclic graphs, trying to remove the less important arcs or to swap them in order to compensate for their loss, until a better valid DAG is found. We use local score changes to guide the search: $\Delta_{Xi}(G, OP) = f_{Xi}(D, G') - f_{Xi}(D, G)$, with $G'$ the result of applying the local move operator $OP \in \{DELETE, SWAP\}$ to $G$. A negative sum of local changes aborts the search. Recall that finding a minimum number of arc deletions in order to restore acyclicity is NP-hard, see *minimum feedback arc set problem* in [10]. We use a greedy approach instead. The pseudo-code of SWAP* is given in Algorithm 2. The local score improvement of the initial SWAP operation is evaluated at line 3. It corresponds to a putative gain on the current score. If it is positive then this operation is applied to a copy of the input DAG $G$, checking next if it creates some directed cycles. Each cycle is retrieved by the `NextCycle` function[7] and the algorithm searches for an arc deletion in this cycle with minimum deterioration of the local score at line 6. If the combined local score change of the SWAP and DELETE operations is positive then it applies the selected arc deletion and continues to test if there are no more directed cycles at line 4. If the combined local score change is negative then it tries to swap an arc of the cycle such that the combined local score change is maximized (line 7) and positive. If it fails to find such an operation then it stops breaking cycles and returns an empty operation set. Finally if it succeeds, breaking all cycles, then it returns a feasible set of SWAP and DELETE operations resulting into a new valid DAG $G'$ with a better score than $G$. The true score improvement is equal to $\Delta$.

Algorithm 2 terminates because there are at most $O(pk)$ directed cycles to break, assuming $p$ nodes and $k$ maximum number of parents per node. Arcs newly created by swap operations cannot be swapped again more than $p$ times thanks to our restricted list of alternative candidate parent nodes $\mathbf{X'}$ used at line 7, initialized to all the variables at the beginning, and updated by the list of nodes present in the current cycle at line 5.

We apply the same approach to extend the other operators ADD* and REVERSE*, breaking cycles with deletions or swaps. Because REVERSE*($X \rightarrow Y$) = ADD*($Y \rightarrow X$), we use only ADD*. The resulting neighborhood exploiting these extended operators is denoted $\mathcal{N}^{A^*DS^*}$ and SGS using this neighborhood is denoted SGS$^3$ (Stochastic Greedy Search with Successive Swaps) in the experiments.

## 4 Experimental Results

In this section, we describe a set of experiments aimed at testing the performance of SGS$^i$ algorithms compared with state-of-the-art Bayesian network structure learning algorithms on standard Bayesian networks and challenging gene regulatory networks.

[7] We implemented an incremental acyclicity test [14] which returns a shortest directed cycle using breadth first search.

## 4.1 Results on Standard Bayesian Networks

We used four gold-standard networks from the Bayesian Network Repository[8] whose main properties are shown in Table 1. In this table, *Nodes* and *Arcs* specify the number of nodes and arcs respectively in the DAG. The *Max in-degree* is the maximum number of parents of a node. The *Min-max states* are the minimum and maximum number of states in the domain of a variable associated to a node. Finally, *Longest path* is the length of the longest directed path between any pair of nodes in the DAG. 100 samples of size $n = 500$ and $n = 5000$ were generated for each network using Causal Explorer [1].

**Table 1.** Bayesian network properties

| | Alarm | Insurance | Hailfinder | Pigs |
|---|---|---|---|---|
| Nodes | 37 | 27 | 56 | 441 |
| Arcs | 46 | 52 | 66 | 592 |
| Max in-degree | 4 | 3 | 4 | 2 |
| Min-max states | 2-4 | 2-5 | 2-11 | 3-3 |
| Longest path | 11 | 10 | 14 | 6 |

We compared SGS$^i$ algorithms with LAGD [17], available in the WEKA software [15] and GES [3] implemented in Tetrad 4.4.0 [24]. LAGD was shown to outperform repeated hill-climbing, order-based search, tabu search, and simulated annealing in [23]. GES, which is considered the reference algorithm for Bayesian network structure learning, was reported having similar performance to the most recent order-based search algorithms in [2]. Recall that SGS$^1$ is similar to repeated hill-climbing, SGS$^2$ uses the SWAP operator, and SGS$^3$ breaks cycles by successive DELETE and SWAP operators. Experiments were performed on a 3 GHz Intel Core2 computer with 4GB running Linux 2.6.

We fixed the maximum number of parents per node at $k = 5$ for SGS$^i$ and LAGD. LAGD exploits a $h = 2$-look ahead in $l = 5$ maximum improving directions. GES was restricted on the number of adjacent nodes: $d = 7$ for Hailfinder and $d = 10$ for Pigs network as done in [2]. All methods were initialized by an empty graph and optimized the BDeu score with *equivalent sample size* $\alpha = 1$ and no prior on the network structures. For each sample, we recorded the best score obtained by GES, and by $r = 10$ randomized greedy searches for SGS$^i$ (Algorithm 1 at line 2) as for LAGD[9].

In order to test the statistical significance of the difference in BDeu score between two methods, we applied a non-parametric paired test, the Wilcoxon signed-rank test [32]. Table 2 presents the test results for all the pairs of methods by using an unilateral alternative (no difference versus better), the pairwise type I error is fixed to $6.2510^{-4}$ which corresponds to a familywise error rate of 5% with Bonferroni correction for multiple comparisons. The above results were summarized in Table 3, in a summarized Wilcoxon score for each method, obtained by summing the positive comparisons and subtracting the negative ones. The non significant comparisons were ignored.

SGS$^3$ was the best method for the four networks, except for Pigs network with $n = 5000$ which is more accurately estimated by GES. We conjecture that in this case, GES was closed to its asymptotic optimal behavior, which may be due to the smallest network features of Pigs network among all. SGS$^2$ improved over SGS$^1$ and reached the second position especially for small sample sizes, probably for

[8] `http://www.cs.huji.ac.il/site//labs/compbio/Repository/`

[9] We randomly permute the input variables at each run.

**Table 2.** Wilcoxon test comparing pairs of algorithms (familywise error rate = 5%). For Method1 versus Method2, + means that Method1 is significantly better than Method2, − means that Method1 is significantly worse than Method2 and ∼ means there is no significant result

| | Alarm | | Insurance | |
|---|---|---|---|---|
| Sample size | 500 | 5 000 | 500 | 5 000 |
| SGS$^3$ vs SGS$^1$ | + | + | + | + |
| SGS$^3$ vs SGS$^2$ | + | + | + | + |
| SGS$^2$ vs SGS$^1$ | + | ∼ | ∼ | ∼ |
| SGS$^3$ vs GES | + | + | + | + |
| SGS$^3$ vs LAGD | + | + | + | + |
| SGS$^2$ vs GES | + | ∼ | + | + |
| SGS$^2$ vs LAGD | ∼ | ∼ | + | + |
| SGS$^1$ vs GES | + | ∼ | + | + |
| SGS$^1$ vs LAGD | ∼ | ∼ | + | + |
| LAGD vs GES | + | ∼ | + | + |
| | Hailfinder | | Pigs | |
| SGS$^3$ vs SGS$^1$ | ∼ | + | + | + |
| SGS$^3$ vs SGS$^2$ | ∼ | + | + | + |
| SGS$^2$ vs SGS$^1$ | ∼ | ∼ | ∼ | ∼ |
| SGS$^3$ vs GES | + | + | + | - |
| SGS$^3$ vs LAGD | ∼ | + | n/a | n/a |
| SGS$^2$ vs GES | + | + | ∼ | - |
| SGS$^2$ vs LAGD | ∼ | + | n/a | n/a |
| SGS$^1$ vs GES | + | + | ∼ | - |
| SGS$^1$ vs LAGD | ∼ | + | n/a | n/a |
| LAGD vs GES | + | + | n/a | n/a |

**Table 3.** Summarized Wilcoxon scores (the higher the score, the better the method)

| | Alarm | Insurance | Hailfinder | Pigs |
|---|---|---|---|---|
| SGS$^3$ | 8 | 8 | 5 | 4 |
| SGS$^2$ | 0 | 2 | 2 | -3 |
| SGS$^1$ | -2 | 2 | 2 | -3 |
| LAGD | -1 | -4 | -1 | n/a |
| GES | -5 | -8 | -8 | 2 |

**Table 4.** Number of spurious edges (+) and missing edges (-) to sum for the structural Hamming distance. Both scores are in bold when giving the best distance per configuration

| | | Alarm | | Insurance | |
|---|---|---|---|---|---|
| Sample size | | 500 | 5 000 | 500 | 5 000 |
| SGS$^3$ | + | **8** | 6 | **4** | **2** |
| | – | **3** | 2 | **20** | **8** |
| LAGD | + | 11 | 8 | **4** | 5 |
| | – | 4 | 2 | **20** | 11 |
| GES | + | **6** | **4** | 2 | 3 |
| | – | **5** | **2** | 23 | 12 |
| | | Hailfinder | | Pigs | |
| SGS$^3$ | + | 17 | **16** | 32 | 41 |
| | – | 24 | **13** | 0 | 0 |
| LAGD | + | 21 | 20 | n/a | n/a |
| | – | 26 | 19 | n/a | n/a |
| GES | + | **15** | 11 | **2** | **0** |
| | – | **24** | 22 | **7** | **0** |

reasons commented in Figure 3. LAGD got poor results, the difference with $SGS^1$ can be explained by a better randomization in $SGS^1$ (Algorithm 1 at line 1). LAGD failed on the Pigs network due to the large number of variables $p = 441$ that makes the exploration of 2-look ahead neighborhoods infeasible in a reasonable time. GES was the worst method of this evaluation (except for Pigs) due to limited sample sizes.

Although the algorithms are designed to maximize a (BDeu) score, we generally look for a network structure as close as possible to the true one. When the sample size grows large, improving the BDeu score leads to reduce the *structural Hamming distance* (SHD) between the learned and the true network. We report in Table 4 the means over 100 datasets (rounded values to the nearest integer) of the missing and spurious edges without taking into account the edge orientations. The SHD is the sum of the above values. We do not report $SGS^1$ and $SGS^2$ results that are outperformed by $SGS^3$. $SGS^3$ (resp. GES) got the best SHD in 4 (resp. 5) configurations and outperformed LAGD (which performed as $SGS^3$ in 1 configuration). GES performed extremely well on the Pigs network, finding the true network with 5,000 samples, whereas $SGS^3$ learned too many edges but recovered all the true edges (even with $n = 500$). The spurious edges learned by $SGS^3$ are exclusively due to random orientations of compelled arcs in v-structures (see Figure 1). Assuming $X1 \rightarrow X3 \leftarrow X2$ in the true network (v-structures are very frequent in the Pigs network) and a large sample size, if during its greedy search $SGS^3$ adds first $X1 \leftarrow X3$ and $X3 \rightarrow X2$ then it will add next a *covering edge* $X1 \rightarrow X2$ or $X1 \leftarrow X2$ in order to find a minimal independence map (see Proposition 1). We corrected this fault by comparing for each covered v-structure the 3 possible non-covered v-structures independently from the other variables and selecting the highest score configuration. After this post-processing step on Pigs network, we had only 1 (resp. 2) spurious edges for $SGS^3$ with 500 (resp. 5000) samples without increasing the missing edges number. The same post-processing had no effect on the other smaller networks.

## 4.2 Detailed analysis on the Alarm network

We conducted a series of algorithm analyzes on the Alarm network with a fixed sample size $n = 500$.

**Table 5.** Single greedy search analysis on the Alarm network ($n = 500, r = 1$)

| | BDeu score | Iter. | BDeu cache | Time(s) |
|---|---|---|---|---|
| $SGS^3$ | -5490.53 | 66 | 4543 | 3.2 |
| $SGS^2$ | -5513.89 | 58 | 4310 | 2.3 |
| $SGS^1$ | -5541.55 | 55 | 3305 | 1.5 |
| LAGD | -5544.61 | 35 | 4782 | 3.2 |
| GES | -5659.26 | 72 | 5531 | 2.4 |

Table 5 shows a typical example of the BDeu score reached by the algorithms for a single greedy search ($r = 1$) on a particular sample. It also reports the number of local moves *Iter.* (Algorithm 1 at line 1), the number of local score $f_{Xi}(D, G)$ computations for *different* parent configurations, *i.e.* assuming a *perfect BDeu cache*, and the CPU time in seconds. As expected, the number of iterations for LAGD was less than half that of other methods, since LAGD applies two local operators (ADD, DELETE, or REVERSE) at each move. The swap operations used by $SGS^2$ and $SGS^3$ slightly increased the number of moves allowing to find better local optima. Comparing CPU times is a difficult task due to the various implementation details (language choice, caching strategy, sample compilation techniques, see for instance Chapter 18.4.3 in [18]). Nevertheless $SGS^i$ algorithms are comparable. In our implementation, $SGS^3$ was twice slower than $SGS^1$. A fairer comparison is the number of calls to new local score computations. $SGS^1$ needed the fewest number, whereas GES needed the most as it tends to explore larger neighborhoods in the space of Markov-equivalent networks. Notice that $SGS^2$ and $SGS^3$ required less score computations than LAGD due to graph sparsity (see Section 3.1). The greater BDeu cache size of $SGS^3$ compared to $SGS^2$ is mainly due to the increased number of local moves.

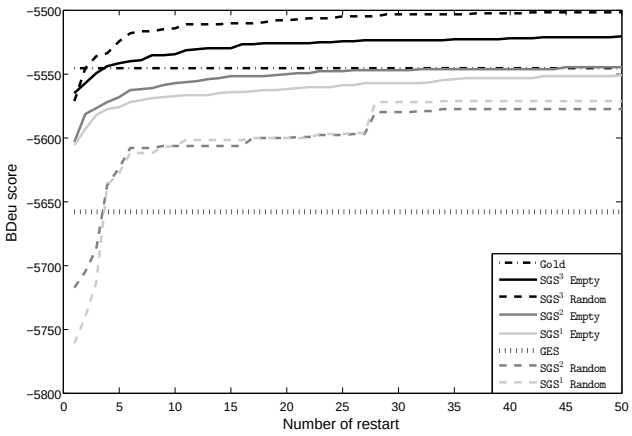



**Figure 5.** Best BDeu scores, averaged on 30 Alarm samples ($n = 500$), found by $SGS^i$ algorithms as the number of restarts $r$ increases and starting either from an empty (solid line) or a random graph (dashed line). Results of GES (dotted line) and BDeu score of the true network *Gold* (dash-dotted line) are also given. Methods are sorted from the best, $SGS^3$ with an empty graph, to the worst, $SGS^1$ with a random initial graph, at $r = 1$.

We further analyzed the impact on performances of the number of restarts $r$ and the initial graph used by $SGS^i$ algorithms (see `InitGraph` in Algorithm 1). Figure 5 reports averaged BDeu scores on 30 Alarm samples (the length of the 95% confidence interval of the BDeu score mean was around ±30). The 30 initial random graphs, common for $SGS^i$ algorithms, are composed of 71 arcs with at most two parents per node[10]. All $SGS^i$ methods reached a better BDeu score than GES in this small sample size situation by the use of less than $r = 4$ restarts. $SGS^i$ methods converged quite rapidly as $r$ increases. Only $SGS^3$ found a better score than the true network. Initial random graphs were counter-productive for all the methods, except for $SGS^3$. It shows that starting from a random graph is useful if the available operators allow to move efficiently in the search space, which is illustrated by the following experiment. On the contrary, using randomness within the greedy selection process (see `SelectRandom` in Algorithm 1) was always beneficial.

Finally, we analyzed how often our new local move operators are used during the search. Table 6 shows the mean number of local moves using a specific operator averaged on $r = 50$ greedy searches over 30 Alarm samples ($n = 500$). Labels ADD+ and SWAP+ mean that at least two local operators were successively applied at a given move in order to break cycles[11]. The new local move operators are

[10] For each node, we randomly select two parents and remove a parent if it creates a directed cycle.

[11] We found at most 4 (resp. 5) successive operations for the 1500 runs starting from an empty (resp. random) graph.

**Table 6.** Operator usage depending on the initial graph

| | Empty graph init. | | | Random graph init. | | |
|---|---|---|---|---|---|---|
| | $SGS^1$ | $SGS^2$ | $SGS^3$ | $SGS^1$ | $SGS^2$ | $SGS^3$ |
| ADD | 53 | 53 | 52 | 53 | 9.5 | 11.5 |
| DELETE | 0.5 | 0.1 | 1.4 | 64 | 20 | 43 |
| REVERSE | 0.9 | 1.1 | 0.7 | 5.3 | 3.4 | 1.9 |
| SWAP | 0 | 1.1 | 3.2 | 0 | 47 | 33 |
| ADD+ | 0 | 0 | 2 | 0 | 0 | 14 |
| SWAP+ | 0 | 0 | 1.6 | 0 | 0 | 9.5 |

mostly used when starting from a random graph (there are more cycles to break), but when starting from an empty graph, only a few applications of the extended moves allow to significantly improve the BDeu score as previously shown in Figure 5 and Table 3.

## 4.3 Results on Gene Regulatory Networks

Gene regulatory network reconstruction from gene expression data using Bayesian network structure learning was first proposed in [11]. We used simulated expression datasets of the Systems Genetics Challenge A from the international reverse enginnering competition DREAM5 [9]. Genetics data were not used as they require additional modelling to be taken into account, see *e.g.* [31]. Expression data were generated using the SysGenSIM generator [22] based on ordinary differential equation simulation. Five datasets are available for three different sample sizes ($n$ = 100, 300, and 999). The 15 datasets were obtained from *different* known gene networks composed of $1,000$ variables and containing directed cycles. For each sample size, the five network structures contain a different number of edges varying from $\approx 2,000$ (*Net1*) to more than $5,000$ (*Net5*). We discretized gene expression levels into 2 to 4 states using an adaptive method based on an adapted $k$-means algorithm and the more general framework of Gaussian mixture models as described in [31].

With such large networks, we had to adapt the learning procedure of $SGS^i$ algorithms[12]. We decided to restrict their lists of candidate parents as done in [13]: we selected for each variable $X$ the set of parents $\mathbf{S}$ such that each element $Y$ of $S$ improves the local BDeu score when it is considered as a unique parent compared to the orphan situation ($f_X(D, \{Y \rightarrow X\}) > f_X(D, \emptyset)$). This filtering process was done prior to the search[13]. In these experiments, $SGS^i$ algorithms have a maximum number of parents per node fixed at $k = 5$ and use $r = 10$ restarts. Instead of LAGD (which was too slow), we used MMHC [30] having two steps similar to $SGS^i$. It first selects the skeleton of the graph using mutual information measures (MMPC [29] filtering step) and then orientates edges in a greedy manner. We recorded the best BDeu score of 10 runs for MMHC, by randomly permuting the input variables at each run. The skeleton being built locally for each variable, MMHC can handle large networks. All the methods started from an empty graph and optimized the BDeu score with $\alpha = 1$ and no prior on the network structures.

It was difficult to perform statistics with the results of this experiment. Indeed, contrary to the standard network experiment, there were no replicated samples of the same network. We decided to pool results per sample size and we performed the Wilcoxon test on the groups. With only five results per group to compare the methods, we are aware that the power of the test is very low. So, we applied a pairwise type I error of 5% and we did not try to correct for multiple comparisons, see Table 7. However, it is worth noting that $SGS^3$ was always the best method and that it increased the BDeu score by about 2% in average.

[12] GES managed in ~1-hour CPU time each network thanks to its better implementation of caching and heap data structure.

[13] It could also be done during the search as in [12].

**Table 7.** Wilcoxon test (error rate = 5%) for different gene network sample sizes

| | 100 | 300 | 999 |
|---|---|---|---|
| $SGS^3$ vs MMHC | + | + | + |
| $SGS^3$ vs GES | + | + | + |
| MMHC vs GES | - | ~ | + |

Surprisingly, GES appeared to be better on smaller sample sizes compared to MMHC. As explained below, MMHC was penalized by its filtering process, especially on the smallest sample size, whereas GES had no restrictions on the candidate parent sets.

In these experiments, the structural Hamming distance (SHD) was not informative due to the poor results reached by all tested algorithms for such large networks. SHD, equal to the sum of spurious edges ($s$) and missing edges ($m$), was greater than the number of edges ($e$) of the true graph. In this situation, even the empty structure appears better. For this reason, we computed another aggregated structural quality measure based on the *precision* ($\frac{e-m}{e-m+s}$) and *recall* ($\frac{e-m}{e}$) measures and took the Euclidean distance to the origin to combine them ($\sqrt{precision^2 + recall^2}$). Contrary to SHD, a high distance indicates a better structural quality. We observed in Table 8 contrasted performances between the tested methods depending on the sample size: for n=100, MMHC got the best results, for $n = 300$, it was GES, and finally $SGS^3$ performed the best for the largest sample size. Better BDeu scores are not always synonymous with a better structural quality, the limited sample size in addition to the non faithfulness of the data (generated by ordinary differential equations with cycles) could explain this behavior. We should notice that the good optimization behavior of $SGS^3$ in terms of BDeu scores resulted in a better structural quality than GES and MMHC as sample size increases.

**Table 8.** Euclidean distances to the origin of the (precision, recall) values. Best distances per sample size are in bold

| $n$ | | $SGS^3$ | MMHC | GES |
|---|---|---|---|---|
| 100 | *Net1* | 0.170 | **0.218** | 0.196 |
| | *Net2* | 0.213 | **0.295** | 0.232 |
| | *Net3* | 0.214 | **0.266** | 0.236 |
| | *Net4* | 0.201 | **0.265** | 0.214 |
| | *Net5* | 0.206 | **0.247** | 0.243 |
| 300 | *Net1* | **0.510** | 0.483 | 0.464 |
| | *Net2* | 0.342 | 0.337 | **0.385** |
| | *Net3* | 0.484 | 0.488 | **0.505** |
| | *Net4* | 0.453 | 0.478 | **0.498** |
| | *Net5* | 0.419 | 0.397 | **0.428** |
| 999 | *Net1* | **0.578** | 0.537 | 0.549 |
| | *Net2* | **0.581** | 0.510 | 0.505 |
| | *Net3* | 0.454 | 0.441 | **0.484** |
| | *Net4* | **0.476** | 0.450 | **0.476** |
| | *Net5* | **0.479** | 0.471 | 0.458 |

Moreover, we compared the quality of the filtering process used by MMPC and $SGS^3$. Table 9 reports for the first network (*Net1*) of each sample size, the total number of arcs kept by the filtering process and the recall value, which represents the percentage of true edges in the filter. Our first observation is the poor recall of both filtering processes, which indicates strong structural differences between the

true network and the learned network even with $n = 999$ sample size. Our filtering approach obtained better recall values with similar compression sizes than MMPC.

Finally, we tried the same methods as in Section 4.1 on 50 small gene regulatory networks (*Web50* Artificial Gene Networks with 50 nodes and 50 arcs) without doing any filtering process. For a sample size $n = 500$, SGS$^3$ was significantly better than GES and LAGD in terms of BDeu scores and slightly better in terms of Euclidean distances to the origin of the (precision, recall) values.

**Table 9.** Total size and recall of candidate parent lists for SGS$^3$ and MMPC on the most sparse gene network *Net1*

| | | 100 | 300 | 999 |
|---|---|---|---|---|
| BDeu test | size | 2670 | 2430 | 5984 |
| | recall | 9% | 18% | 35% |
| MMPC | size | 2568 | 3064 | 3842 |
| | recall | 5% | 12% | 23% |

## 5 Conclusion

We have presented in this paper a new greedy search algorithms called SGS$^i$ exploiting stochasticity from two random draws. We have developed a new local move operator called SWAP and extended versions for ADD and SWAP operators to overcome frequent limitations of local search methods which are local maxima and cyclic situations. We compared SGS$^3$ using SWAP and extended operators to state-of-the-art methods and we obtained significant BDeu and recall value improvements on classical benchmarks when the sample size is small. The complexity of SGS$^3$ stays moderate with sparse networks. In case of large networks with many variables we applied a filtering process in preprocessing using the same criterion as for the search. This process kept more edges from the true network than mutual information-based methods with significant reduction of the search space.

In the future, we would like to test our new operators with other local search methods like tabu search.

## REFERENCES


[1] C. Aliferis, I. Tsamardinos, A. Statnikov, and L. Brown, 'Causal Explorer: A Probabilistic Network Learning Toolkit for Biomedical Discovery', in *Proceedings of the International Conference on Mathematics and Engineering Techniques in Medicine and Biological Sciences*, (2003).

[2] J. Alonso-Barba, L. de la Ossa, and J. Puerta, 'Structural learning of bayesian networks using local algorithms based on the space of orderings', *Soft Comput*, 1881–1895, (2011).

[3] D. Chickering, 'Optimal structure identification with greedy search', *Journal of Machine Learning Research*, **3**, 507–554, (2002).

[4] D. Chickering, D. Heckerman, and C. Meek, 'Large-Sample Learning of Bayesian Networks is NP-Hard', *Journal of Machine Learning Research*, **5**, 1287–1330, (2004).

[5] D. Chickering and D. Heckermann, 'Learning bayesian networks is NP-complete', *In learning from data: Al and Statistics*, (1996).

[6] G Cooper and E Hersovits, 'A Bayesian method for the induction of probabilistic networks from data', *Machine Learning*, **9**, 309–347, (1992).

[7] R. Daly and Q. Shen, 'Learning Bayesian Network Equivalence Classes with Ant Colony Optimization', *Journal of Artificial Intelligence Research*, **35**, 391–447, (2009).

[8] R. Daly, Q. Shen, and S. Aitken, 'Learning Bayesian networks: approaches and issues', *The Knowledge Engineering Review*, **26**(2), 99–157, (2011).

[9] 'The DREAM5 Systems Genetics Challenges'. http://wiki.c2b2.columbia.edu/dream/index.php/D5c3, 2010.

[10] Paola Festa, Panos M. Pardalos, and Mauricio G. C. Resende, 'Feedback Set Problems', in *Encyclopedia of Optimization*, 1005–1016, Springer, (2009).

[11] N. Friedman, M. Linial, I. Nachman, and D. Pe'er, 'Using bayesian networks to analyse expression data', *Journal of computational biology*, **7**(3/4), 601–620, (2000).

[12] J. Gámez, J. Mateo, and J. Puerta, 'Learning Bayesian networks by hill climbing: efficient methods based on progressive restriction of the neighborhood', *Data Min. Knowl. Discov.*, **22**, 106–148, (2011).

[13] A. Goldenberg and A. Moore, 'Tractable learning of large Bayes net structures from sparse data', in *Proc. of ICML'04*, pp. 44–51, (2004).

[14] B. Haeupler, T. Kavitha, R. Mathew, S. Sen, and R. Tarjan, 'Faster algorithms for incremental topological ordering', in *Proc. of ICALP*, pp. 421–433, (2008).

[15] M. Hall, F. Eibe, G. Holmes, B. Pfahringer, P. Reutemann, and I. Witten, 'The WEKA Data Mining Software', *SIGKDD Explorations*, **11**, (2009).

[16] D. Heckerman, D. Geiger, and D. Chickering, 'Learning Bayesian Networks: The Combination of Knowledge and Statistical Data', in *Machine Learning*, volume 20, pp. 197–243, (1995).

[17] A. Holland, M. Fathi, M. Abramovici, and M. Neubach, 'Competing fusion for bayesian applications', in *Proc. of IPMU 2008*, pp. 378–385, (2008).

[18] D. Koller and N. Friedman, *Probabilistic Graphical Models: Principles and Techniques*, MIT Press, 2009.

[19] G. Melançon, I. Dutour, and M. Bousquet-Mélou, 'Random generation of directed acyclic graphs', *Electronic Notes in Discrete Mathematics*, **10**, 202–207, (2001).

[20] A. Moore and W.K. Wong, 'Optimal reinsertion: A new search operator for accelerated and more accurate bayesian network structure learning', in *Proc. of ICML '03*, pp. 552–559, (2003).

[21] J. Nielsen, T. Kocka, and J. Pefia, 'On Local Optima in Learning Bayesian Networks', in *Proc. of UAI-03*, pp. 435–442, (2003).

[22] A. Pinna, N. Soranzo, I. Hoeschele, and A. de la Fuente, 'Simulating system genetics data with SysGenSIM', *Bioinformatics*, **27**, 2459–2462, (2011).

[23] E. Salehi and R. Gras, 'An empirical comparison of the efficiency of several local search heuristics algorithms for Bayesian network structure learning', in *Learning and Intelligent OptimizatioN Workshop (LION 3)*, (2009).

[24] R. Scheines, P. Spirtes, C. Glymour, C. Meek, and T. Richardson, 'The TETRAD Project: Constraint Based Aids to Causal Model Specification', *Multivariate Behavioral Research*, **33**(1), 65–117, (1998).

[25] M. Schmidt, A. Niculescu-Mizil, and K. Murphy, 'Learning graphical model structure using L1-regularization paths', in *Proc. of AAAI'07*, pp. 1278–1283, (2007).

[26] G. Schwarz, 'Estimating the dimension of a model', *The annals of statistics*, (1978).

[27] P. Spirtes, C. Glymour, and R. Scheines, 'Causality from probability', *Evolving knowledge in the natural and behavioral sciences*, 181–199, (1990).

[28] M. Teyssier and D. Koller, 'Ordering-based Search: A Simple and Effective Algorithm for Learning Bayesian Networks', in *Proc. of UAI'05*, pp. 584–590, (2005).

[29] I. Tsamardinos, C. Aliferis, and A. Statnikov, 'Time and sample efficient discovery of Markov blankets and direct causal relations', in *Proc. of KDD'03*, pp. 673–678, (2003).

[30] I. Tsamardinos, L. Brown, and C. Aliferis, 'The max-min hill-climbing Bayesian network structure learning algorithm', *Mach. Learn.*, **65**, 31–78, (2006).

[31] M. Vignes, J. Vandel, D. Allouche, N. Ramadan-Alban, C. Cierco-Ayrolles, T. Schiex, B. Mangin, and S. de Givry, 'Gene Regulatory Network Reconstruction Using Bayesian Networks, the Dantzig Selector, the Lasso and Their Meta-Analysis', *PLoS ONE*, **6**, (2011).

[32] F. Wilcoxon, 'Individual Comparisons by Ranking Methods', *Biometrics Bulletin*, **1**, 80–83, (1945).

# Comparative Study for Inference of Hidden Classes in Stochastic Block Models

**Pan Zhang**[1] and **Florent Krzakala**[2] and **Jörg Reichardt**[3] and **Lenka Zdeborová**[4]

**Abstract.** Inference of hidden classes in stochastic block model is a classical problem with important applications. Most commonly used methods for this problem involve naïve mean field approaches or heuristic spectral methods. Recently, belief propagation was proposed for this problem. In this contribution we perform a comparative study between the three methods on synthetically created networks. We show that belief propagation shows much better performance when compared to naïve mean field and spectral approaches. This applies to accuracy, computational efficiency and the tendency to overfit the data.

## 1 Introduction

A large portion of the intriguing emergent phenomena of complex many particle systems is a consequence of the structure of interactions among their constituents. Bluntly, a soup of neurons does not have the same capabilities as a specifically woven neural net. Similar considerations apply to social systems, information systems, biological systems or economical systems where the patterns of interaction are far from random and result in complex system-wide phenomena.

Fueled by a flood of readily available relational data, recent years have seen a surge of research focused on structural properties of networks as first step to understanding some of the properties of complex systems and ultimately their function [5, 17].

Interestingly, it is often much easier to map the network of interactions than to explain its function. A prime example of this phenomenon are protein interaction networks. Modern biotechnology allows to automatize charting the matrix of pairwise binding relations for all proteins produced by an organism, *i.e.* do two proteins form a stable link or not [23]. As proteins generally operate in complexes (agglomerates of several proteins) such a network of pairwise interactions encodes latent information about protein function. Hence, it makes sense to use network structure to make inferences about protein function or plan and guide other wet-lab experiments aimed at elucidating function [19]. Similar considerations apply to the analysis of social networks where interactions are recorded in online data streams but information on the properties of the actual agents remains hidden behind pseudonyms or avatars [25].

Hence, the hypothesis behind network analysis is that nodes in a network which have similar patterns of interaction are likely to have common properties or perform similar function. Discovering topological similarities and differences thus hints at the existence of possible latent features of the nodes in the network that merit further analysis.

Being a first step to more detailed analysis, such exploratory analysis is often highly consequential. It is important to thoroughly understand the algorithms used in every detail and to be aware of possible limitations and pitfalls. This contribution aims at raising this awareness using the simple example of inferring the parameters of a Poisson-mixture model, the so-called Stochastic-Block-Model (SMB) [9, 6], in undirected unweighted unipartite networks. The conclusions we draw, however, extend well beyond this example and we discuss these consequences at the end of the paper.

Our contribution is then organized as follows: first we introduce the stochastic block model as a way to capture density fluctuations in relational datasets and infer latent variables. Next, we discuss the theoretical limitations that any inference technique for such a model must face: namely a sharp transition between a parameter region where inference is feasible and a parameter region where inference is impossible. Third, we briefly review spectral approaches and the Expectation Maximization (EM) algorithm in conjunction with the naïve mean field approach. We then introduce a formulation of the EM algorithm based on belief propagation. Fourth, we compare the performance of these three approaches on ensembles of benchmark networks from a region near the above mentioned feasible-infeasible transition in the parameter space. In this region, particularly difficult problem instances can be found that allow to highlight performance differences. Finally, we discuss our findings and the possible extensions and consequences to other models and inference tasks.

## 2 The Stochastic Block Model

The simplest model of a network of $N$ nodes and $M$ undirected unweighted edges between them is a an Erdős-Rényi graph. It assumes that a link falls between any pair of nodes $(i, j)$ with constant probability $p_{ij} = p_0 = 2M/[N(N-1)]$, independently of whether links exist between other pairs of

[1] CNRS and ESPCI ParisTech, 10 rue Vauquelin, UMR 7083 Gulliver, Paris 75005,France, email: pan.zhang@espci.fr
[2] CNRS and ESPCI ParisTech, 10 rue Vauquelin, UMR 7083 Gulliver, Paris 75005, France, email: fk@espci.fr
[3] Institute for Theoretical Physics, University of Würzburg Am Hubland 97074 Würzburg, Germany, email:reichardt@physik.uni-wuerzburg.de
[4] Institut de Physique Théorique, IPhT, CEA Saclay, and URA 2306, CNRS, 91191 Gif-sur-Yvette, France, email: lenka.zdeborova@gmail.com

nodes. Consequentially, large networks with low link density $p_0$ generated by this model have a Poissonian degree distribution with mean degree $\langle k \rangle = p_0(N-1)$. This model can already explain two main characteristics of real world networks - their small world property of short average path lengths and their connectedness even at low densities. Unfortunately, it cannot explain much more. In particular, it fails to capture the large variance of link densities between groups of nodes observed in many networks.

In real world networks, not all nodes are created equal and may represent entities of very different properties or functions. Whether two nodes are linked often depends on these properties. Consider the example of protein interaction again. Membrane proteins will certainly bind to other membrane proteins to form stable membranes, but, for example, the enzymes involved in various catalytic reactions should not stick to the cell membrane since otherwise the interior of the cell would soon be depleted of these essential molecules [19]. In an entirely different social context, one will certainly observe social interactions correlated with the agents' age, gender, possibly income or education. Social ties will depend on these qualities an thus network structure is indicative of node properties and may be used to make corresponding inferences.

One of the simplest models capable of capturing the dependence of link probability on node properties is the Stochastic Block Model [9]. It assumes that each node $i \in \{1, .., N\}$ is of one and only one of $q$ classes and $t_i = r$ is indicating the membership of node $i$ in class $r \in \{1, .., q\}$. As before, nodes are linked independently, but now, the probability of node $i$ linking to node $j$ depends on $t_i$ and $t_j$ alone, *i.e.* $p_{ij} = p_{t_i t_j}$. One can easily write down a probabilistic generative model for this sort of network. First, we assume that nodes are assigned into $q$ classes randomly by a multinomial distribution with parameters $\mathcal{P}(t_i = r) = p_r$. Next, we specify the matrix of link probabilities between classes $p_{rs} \in (0,1)^{q \times q}$. Our set of parameter thus comprises of $\theta = \{q, p_r, p_{rs}\}$. The probability of generating a specific $\{0,1\}^{N \times N}$ adjacency matrix $\mathbf{A}$ together with a specific assignment of nodes into classes $\mathbf{t}$ is then given as:

$$\mathcal{P}(\mathbf{A}, \mathbf{t}|\theta) = \prod_{i<j} \left[ p_{t_i t_j}^{A_{ij}} (1 - p_{t_i t_j})^{(1-A_{ij})} \right] \prod_i p_{t_i} \qquad (1)$$

The expected average density of links in such a network is $p_0 = \sum_{rs} p_r p_{rs} p_s$. If we were able to observe the adjacency matrix $\mathbf{A}$ *and* class memberships $\mathbf{t}$ at unknown parameters, equation (1) would give us the complete data likelihood of the parameters $\theta$. It is then easy to estimate the parameters $\theta^*$ which maximize (1):

$$\begin{aligned} p_r &= \frac{1}{N} \sum_i \delta_{t_i,r} \qquad (2) \\ p_{rs} &= \frac{1+\delta_{rs}}{N p_r (N p_s - \delta_{rs})} \sum_{i<j} A_{ij} \delta_{t_i,r} \delta_{t_j,s} \end{aligned}$$

With (1) being a member of the exponential family, these estimators are consistent, efficient and the model is identifiable, *i.e.* the maxima are unique. In this contribution we always assume that the correct number of classes $q$ is known.

However, in practical applications as discussed, the situation is often that we only have access to the adjacency matrix $\mathbf{A}$ but *not* to the class labels $\mathbf{t}$ which are our primary interest for explaining network structure and possibly function. Fortunately, under certain circumstances we can still draw conclusions about these hidden variables using the toolbox of statistical inference. What these circumstances are and how this is usually done will be discussed in the following two sections.

## 3 General Considerations

It is clear that the task of inferring the unobserved latent variables is only possible if the preference matrix $p_{rs}$ shows sufficient "contrast". If all entries were the same, *i.e.* $p_{rs} = p_0$, then of course no method can perform any inference on the hidden variables. Conversely, if $p_{rs} = p_0 \delta_{r,s}$, then the network practically consists of several disconnected components and inference reduces to the trivial task of identifying the component to which an individual node belongs. Between these two extremes of impossible and trivial, there is a sharp phase transition [20, 3, 2]. It divides the parameter space into a region where it is provably impossible to infer the latent variables with an accuracy higher than guessing and a region where it is possible with high accuracy.

Theoretical analysis has shown that the transition persists in infinitely large networks when they are sparse, *i.e.* the average degree per node does not grow with the system size. In other words, networks in which the elements of $p_{rs}$ scale as $1/N$. In contrast, for dense networks in which $p_{rs}$ does not scale with $N$, considering larger networks means considering proportionally larger average degrees and this will render even very small amounts of variance in $p_{rs}$ detectable and thus lets the region of impossible inference vanish [18].

In real applications, we cannot generally increase network size at constant parameters. We will observe both the region of impossible and possible inference. However, the parameter region of impossible inference will be smaller for denser networks, *i.e.* those with higher average degree. Further, it has been shown that networks with parameters in the vicinity of the transition point are the instances in which inference is hardest [3, 2].

As it is our aim to highlight performance differences between different inference techniques for the SBM, we will focus our attention on instances in sparse graphs near the transition from impossible to possible inference. Before we come to this analysis, however, we will introduce the contestants.

## 4 Inferring Stochastic Block models

When inferring latent structure in data, one can take the route of statistical inference if one can justify a statistical model to fit to the data as we've done with the SBM. It may also be sensible to use a simple dimensionality reducing heuristic. We consider both of these approaches.

### 4.1 Spectral Approaches

When dealing with high dimensional data such as networks and searching for common patterns of interactions, a natural strategy is to try reducing the dimensionality in such a way that nodes with similar interaction partners are mapped to positions in some low dimensional space, while nodes with very different interaction partners should be positioned far apart. One then uses standard clustering algorithms, such as $k$-means in our case, originally developed for multivariate data

and to analyze the nodes in their low dimensional embedding. This is the strategy behind all spectral techniques of network analysis.

Let us consider the adjacency matrix $\mathbf{A}$ as a list of $N$ measurements in an $N$-dimensional feature space, each row describing one node in $N$ dimensions, namely, its relations to the other nodes. We could then apply a variant of multidimensional scaling such as principal component analysis (PCA). We would subtract the means of the measurements in each dimension, calculate the covariance matrix and find the directions of maximum variance by an eigen-decomposition of the co-variance matrix. Finally, we would project our data matrix onto the first $q$ principal components, *i.e.* those eigenvectors of the covariance matrix corresponding to the largest eigenvalues.

A method similar in spirit has been introduced specifically for networks [15]. It differs from PCA only slightly in that it not only removes the means of the rows, but, since $\mathbf{A}$ is symmetric, also the means of the columns. This is to say, the original matrix $\mathbf{A}$ is transformed into a so called modularity matrix $\mathbf{B}$ via

$$B_{ij} = A_{ij} - \frac{k_i k_j}{2M}. \tag{3}$$

This modularity matrix $\mathbf{B}$ now has row-sums and column-sums zero. Note that the terms $k_i k_j/2M \ll 1$ for sparse networks. Since $\mathbf{B}$ is symmetric, the eigenvectors of a corresponding "covariance matrix" $\mathbf{C} = \mathbf{B}\mathbf{B}^{\mathrm{T}}$ are the eigenvectors of $\mathbf{B}$ and hence the projection of the modularity matrix onto the "principal components" is given directly by the components of the eigenvectors corresponding to the largest magnitude eigenvectors of $\mathbf{B}$. This approach has recently been claimed to be no worse than any other approach [13] and we will evaluate this claim in this paper.

Another aspect of this method is worth mentioning. It is known that the best rank-$q$ approximation to a symmetric matrix is given by its eigen-decomposition retaining only the $q$ eigenvalues largest in magnitude. "Best" here means in terms of reconstruction error under the Frobenius norm. If $\mathbf{V}$ is a matrix the columns of which are the eigenvectors of $\mathbf{B}$ ordered by decreasing magnitude of the corresponding eigenvalue, then the entries of the optimal rank-$q$ approximation $\mathbf{B}'$ will be given by

$$B'_{ij} = \sum_{r=1}^{q} V_{ir}\lambda_r V_{jr}. \tag{4}$$

So we see that $B'_{ij}$ is large when the rows $i$ and $j$ of $\mathbf{V}$ are parallel *and* all the considered $\lambda_r$ with $r \in \{1, .., q\}$ are positive. In contrast, if all $\lambda_r$ are negative, rows $i$ and $j$ of $\mathbf{V}$ should be anti-parallel to make $B'_{ij}$ large. Large positive eigenvalues are indicative of block models with some $p_{rr}$ large while large negative eigenvalues are indicative of block models with some $p_{rr}$ small in comparison to the average density of the network $p_0$. We can conclude that when these cases mix, it will generally be very difficult to find an embedding that maps nodes from a network with similar interaction patterns to positions that are close in space using spectral decomposition of the modularity matrix.

Instead of using an embedding that minimizes a reconstruction error, one can also introduce a pairwise similarity measure based on the network topology and then find an embedding of the $N \times N$ similarity matrix such that "similar nodes" are "close". This approach is implemented in the widely used diffusion-map [11].

Assume a random walker is traversing the network. When at node $i$, the walker will then move to any node $j \neq i$ with probability $p_{j|i} = A_{ij}/k_i$. Here, $k_i = \sum_j A_{ij}$ is the number of neighbors of node $i$. We can identify in $p_{j|i}$ as the entries of an $N \times N$ row stochastic transition matrix $\mathbf{P} = \mathbf{D}^{-1}\mathbf{A}$ where $\mathbf{D}$ is a diagonal matrix with $D_{ii} = k_i$. The probability that the random walker, after starting in node $i$, reaches node $j$ in exactly $t$ steps is then given as $p_t(j|i) \equiv \mathbf{P}^t_{ij}$. The stationary distribution of the random walker on the $N$ nodes of the network is given by $\pi^i_0 \equiv \lim_{t\to\infty} p_t(i|j) = k_i/2M$. Equipped with these definitions, one can define a "diffusion distance" between nodes $i$ and $j$ via

$$D^2_t(i,j) = \sum_k \frac{(p_t(k|i) - p_t(k|j))^2}{\pi^k_0}. \tag{5}$$

This is a sensible measure of topological distance between nodes $i$ and $j$ as it measures a difference in the distributions of arrival sites when the random walker starts from either $i$ or $j$. One can find an optimal embedding such that the Euclidean distance in the low dimensional space matches the diffusion distance to any desired precision. The coordinates of this embedding are given by the entries in the eigenvectors corresponding to the $q$ largest non-trivial right eigenvectors of $\mathbf{P}$ scaled by the corresponding eigenvalue to power $t$. Since the largest right eigenvalue of $\mathbf{P}$ is always $\lambda_1 = 1$ and the corresponding eigenvector is constant, it is considered trivial. If a match to relative precision $\delta$ is required we must include all eigenvectors $\mathbf{v}_r$ of $\mathbf{P}$ with $|\lambda_r|^t > \delta|\lambda_2|^t$ where the $\lambda$ are the right eigenvalues of $\mathbf{P}$. As all eigenvalues of $\mathbf{P}$ are smaller in magnitude than 1, $\lambda_2$ dominates for large $t$ and thus the large scale structural features. In this case, large negative eigenvalues are not a problem, since the embedding is such that Euclidian distance between the positions of the nodes in the low dimensional space approximates the topological distance and not the scalar product dressed with the eigenvalues as in the case of the spectral decomposition.

## 4.2 Expectation Maximization

The goal of maximum likelihood inference aims to estimate parameters for a generative model such that the observed data becomes maximally likely under this model. Our generative model (1) gives us the probability of observing the network *and* the node classes. If only the network is observed we need to trace out the node classes. Specifically, we seek

$$\theta^* = \mathrm{argmax}_\theta \mathcal{L}(\theta) \equiv \log \sum_{\mathbf{t}} \mathcal{P}(\mathbf{A}, \mathbf{t}|\theta). \tag{6}$$

The sum over all possible assignments of nodes into latent classes is computationally intractable and so one resorts defining a lower bound on the log-likelihood $\mathcal{L}(\theta)$ which can be both evaluated and maximized. This bound is know as the Free Energy

$$\mathcal{F}(\tilde{\mathcal{P}}(\mathbf{t}), \theta) \equiv \sum_t \tilde{\mathcal{P}}(\mathbf{t}) \log \mathcal{P}(\mathbf{A}, \mathbf{t}|\theta) - \sum_t \tilde{\mathcal{P}}(\mathbf{t}) \log \tilde{\mathcal{P}}(\mathbf{t}). \tag{7}$$

The Free energy $\mathcal{F}$ is a functional of a distribution over the latent variables $\tilde{\mathcal{P}}(\mathbf{t})$ and the model parameters $\theta$. It is easily

shown that $\mathcal{F}$ is indeed a lower bound on $\mathcal{L}(\theta)$:

$$\mathcal{F}(\tilde{\mathcal{P}}(\mathbf{t}), \theta) = -D_{\mathrm{KL}}(\tilde{\mathcal{P}}(\mathbf{t})||\mathcal{P}(\mathbf{t}|\mathbf{A}, \theta)) + \mathcal{L}(\theta). \quad (8)$$

and that if $\mathcal{F}$ has a (global) maximum in $(\tilde{\mathcal{P}}^*(\mathbf{t}), \theta^*)$ then $\mathcal{L}(\theta)$ also has a (global) maximum in $\theta^*$ [14]. The procedure for maximizing $\mathcal{F}$ in turn with respect to its two arguments is known as the Expectation Maximization algorithm [4]. Specifically, maximizing $\mathcal{F}$ with respect to $\tilde{\mathcal{P}}(\mathbf{t})$ at fixed $\theta$ is known as the "E-Step", while maximizing $\mathcal{F}$ with respect to $\theta$ at fixed $\tilde{\mathcal{P}}(\mathbf{t})$ is known as the "M-Step". Ideally, the E-step tightens the bound by setting $\tilde{\mathcal{P}}(\mathbf{t}) = \mathcal{P}(\mathbf{t}|\mathbf{A}, \theta)$, but for our model (1) the calculation of $\mathcal{P}(\mathbf{t}|\mathbf{A}, \theta)$ is also intractable. Note that this is in contrast to estimating the parameters of a mixture of Gaussians where, for observed data $\mathbf{X}$, we can easily evaluate $\mathcal{P}(\mathbf{t}|\mathbf{X}, \theta)$.

Two routes of approximation now lie ahead of us: the first one is to restrict ourselves to a simple factorizable form of $\tilde{\mathcal{P}}(\mathbf{t}) = \prod_i \tilde{\mathcal{P}}(t_i)$ which leads to the mean field approach. The second route leads to belief propagation.

### 4.3 E-Step and M-Steps using the naïve mean field

We shall start by the mean field equations as used for the SBM for instance in [1] or [8]. In addition to the assumption of a factorizing $\tilde{\mathcal{P}}(\mathbf{t})$, one introduces the following shorthand: $\psi_r^i \equiv \tilde{\mathcal{P}}(t_i = r)$. Then, the free energy in the naïve mean field approximation is given by

$$\begin{aligned} \mathcal{F}_{\mathrm{MF}} = & \sum_{i<j,rs} \left( A_{ij} \log \tfrac{p_{rs}}{1-p_{rs}} + \log(1-p_{rs}) \right) \psi_r^i \psi_s^j \\ & + \sum_{i,r} \psi_r^i (\log p_r - \log \psi_r^i) \end{aligned} \quad (9)$$

This free energy is to be maximized with respect to the $\psi_r^i$ by setting the corresponding derivatives to zero and we obtain a set of self-consistent equations the $\psi_r^i$ have to satisfy at $\nabla_\psi \mathcal{F} = 0$:

$$\begin{aligned} \psi_r^i &= \frac{p_r e^{h_r^i}}{\sum_s p_s e^{h_s^i}} \quad (10) \\ h_r^i &= \sum_{j \neq i, s} A_{ij} \log \tfrac{p_{rs}}{1-p_{rs}} \psi_s^j + \sum_s (N - \delta_{rs}) p_s \log(1-p_{rs}) \end{aligned}$$

The beauty of this approach is its apparent computational simplicity as an update of $\tilde{\mathcal{P}}(\mathbf{t})$ can be carried out in $\mathcal{O}(N\langle k \rangle q^2)$ steps. Setting $\nabla_\theta \mathcal{F}_{\mathrm{MF}}$ equal to zero and observing the constraint that $\sum_r p_r = 1$, we derive the following equations for the M-step:

$$\begin{aligned} p_r &= \tfrac{1}{N} \sum_i \psi_r^i \quad (11) \\ p_{rs} &= \frac{\sum_{i<j} A_{ij} \psi_r^i \psi_s^j}{\sum_{i<j} \psi_r^i \psi_s^j} \end{aligned}$$

Note the similarities between eqns. (11) and (2).

### 4.4 E-Step and M-Steps using Belief Propagation

Belief propagation equations for mixture models were used by several authors, see e.g. [7, 22, 21]. Several important nuances in the algorithm make us adopt belief propagation algorithm for SBM as developed in [3, 2], the implementation can be dowloaded at http://mode_net.krzakala.org/.

There are several ways one can derive the Belief Propagation equations (see for instance [26]). One way is from a recursive computation of the free energy under the assumption that the graphical model is a tree. Application of the same equations on loopy graphical models is then often justified by the fact that correlations between variables induced by loops decay very fast and are hence negligible in the thermodynamic limit. In the case treated here, even when the adjacency graph $A_{ij}$ is sparse, the graphical model representing the probability distribution (1) is a fully connected graph on $N$ nodes. However, for sparse networks the interaction for nodes that are not connected by an edge is weak $1 - p_{rs} \approx 1$ and the network of strong interactions is locally tree-like. This puts us in the favorable situation of decaying correlations. This was used in [3, 2] to argue heuristically that in the limit of large $N$ the belief propagation approach estimates asymptotically exact values of the marginal probabilities $\psi_r^i$ and of the log-likelihood, in a special case of block model parameters this has been proven rigorously in [12].

To write the belief propagation equations for the likelihood (1) we define conditional marginal probabilities, or *messages*, denoted $\psi_r^{i \to j} \equiv \mathcal{P}(t_i = r|\mathbf{A} \backslash A_{ij}, \theta)$. This is the marginal probability that the node $i$ belongs to group $r$ in the absence of node $j$. In the tree approximation we then assume that the only correlations between $i$'s neighbors are mediated through $i$, so that if $i$ were missing—or if its group assignment was fixed—the distribution of its neighbors' states would be a product distribution. In that case, we can compute the message that $i$ sends $j$ recursively in terms of the messages that $i$ receives from its other neighbors $k$ [3, 2]:

$$\begin{aligned} \psi_r^{i \to j} &= \frac{p_r e^{h_r^{i \to j}}}{\sum_s p_s e^{h_s^{i \to j}}} \quad (12) \\ h_r^{i \to j} &= \sum_{k \neq i, j} \log \left[ \sum_s \left( \tfrac{p_{rs}}{1-p_{rs}} \right)^{A_{ik}} (1-p_{rs}) \psi_s^{k \to i} \right] \quad (13) \end{aligned}$$

The marginal probability $\psi_r^i$ is then recovered from the messages using (10) and

$$h_r^i = \sum_{j \neq i} \log \left[ \sum_s \left( \frac{p_{rs}}{1-p_{rs}} \right)^{A_{ij}} (1-p_{rs}) \psi_s^{j \to i} \right]. \quad (14)$$

Compared with equations (10), updating the belief propagation equations takes $\mathcal{O}(N^2 q^2)$ steps.

Most real world networks, however, are relatively sparse, *i.e.* the number of edges is much smaller than $N^2$. For such cases the BP equations can be simplified. To see this we consider $c_{rs} = N p_{rs} = O(1)$, in the limit $N \to \infty$ terms $o(N)$ can be neglected as in [2], one then needs to keep and update messages $\psi_r^{i \to j}$ only when $A_{ij} = 1$. The update equation for field $h_r^{i \to j}$ then is

$$h_r^{i \to j} = \sum_{k \in \partial i \backslash j} \log \left( \sum_s c_{rs} \psi_s^{k \to i} \right) - \frac{1}{N} \sum_{k=1}^{N} \sum_s c_{rs} \psi_s^k, \quad (15)$$

where $\partial i$ denotes $i$'s neighborhood. In order to get the marginal probability $\psi_r^i$ one uses eq. (10) and

$$h_r^i = \sum_{k \in \partial i} \log \left( \sum_s c_{rs} \psi_s^{k \to i} \right) - \frac{1}{N} \sum_{k=1}^{N} \sum_s c_{rs} \psi_s^k. \quad (16)$$

Note that it is possible to implement the update of all fields $h_r^i$ in $\mathcal{O}(N\langle k\rangle q^2)$ steps, thus making the BP approach as fast the the naïve mean field. In order to do that, we compute the second term in eq. (15) once at the beginning and then we only add and subtract the contributions to this term that changed.

Once the fixed point of BP equations is found, one uses the Bethe formula to compute the free energy [26]

$$\mathcal{F}_{\rm BP} = \frac{1}{N}\sum_{(ij)\in E}\log Z^{ij} - \frac{1}{N}\sum_i \log\left(\sum_s p_s e^{h_s^i}\right) - \frac{\langle k\rangle}{2}\,, \quad (17)$$

where

$$Z^{ij} = \sum_{r,s} c_{rs}\psi_r^{i\to j}\psi_s^{j\to i}$$

Again the Bethe free energy is exact if the graphical model is a tree and is a very good approximation to the true free energy in many practical cases, and often a much better one than the MF free energy. An important point is that the Bethe free energy is not guarantied to be a bound on the log-likelihood.

Setting $\nabla_\theta \mathcal{F}_{\rm BP}$ equal to zero and observing that the BP equations are stationarity conditions for the Bethe free energy, one derives the following equations for the M-step of expectation maximization

$$\begin{aligned} p_r &= \frac{1}{N}\sum_i \psi_r^i\,, \quad (18)\\ c_{rs} &= \frac{1}{N}\frac{1}{p_r p_s}\sum_{(i,j)\in E}\frac{c_{rs}(\psi_r^{i\to j}\psi_s^{j\to i} + \psi_s^{i\to j}\psi_r^{j\to i})}{Z^{ij}}. \end{aligned}$$

## 5 Performance Comparison

We will compare the performance of the three approaches presented in the last section on ensembles of test networks which have been generated from (1). Hence, we know the true assignment of nodes into classes $t_i$ for all nodes $i \in \{1,..,N\}$. Let us denote by $t_i^*$ the estimates of group assignment that follow from the above algorithms. A simple performance measure is then the "overlap" between $\{t_i\}$ and $\{t_i^*\}$ defined as

$$Q \equiv \frac{1}{N}\max_\pi \sum_i \delta(t_i^*, \pi(t_i)). \quad (19)$$

Since the $t_i$ can only be recovered up to permutation of the class labels, the maximum over all possible permutations of $\pi$ on $q$ elements is taken. Note that a trivial estimate would be $t_i^* = \mathrm{argmax}_r p_r\, \forall i$. Hence, only values of $Q > \max_r p_r$ should be considered as successful inference.

### 5.1 Belief Propagation vs Mean Field

To make a comparison of BP and MF we will assume in both approaches that the parameters $p_r$, $p_{rs}$, and the right number of groups $q$ are known. Both approaches output the estimates of marginal probabilities $\psi_r^i$. In order to estimate the original group assignment, we assign to each node its most-likely group, i.e.

$$t_i^* = \mathrm{argmax}_r \psi_r^i\,. \quad (20)$$

If the maximum of $\psi_r^i$ is not unique, we choose at random from all the $q_i$ achieving the maximum. We refer to this method of estimating the groups as *marginalization*. Indeed, a standard result show that it is the optimal estimator of the original group assignment $\{t_i\}$ if we seek to maximize the number of nodes at which $t_i = t_i^*$.

In practical situations, when the true assignment is not known, one can also use the estimates of the marginal probabilities $\psi_r^i$ to compute the confidence of the method about the estimate $t_i^*$ defined as

$$C \equiv \frac{1}{N}\sum_i \psi_{t_i^*}^i\,. \quad (21)$$

An important remark is that if the marginals $\psi_r^i$ were evaluated exactly then in the large $N$ limit the overlap and confidence quantities agree, $C = Q$. In our tests the quantity $C - Q$ hence measures the amount of illusive confidence of the method. Values of $C - Q$ larger than zero are very undesirable as they indicate a misleading correlation, and give an illusive information on the amount of information reconstructed.

To compare the performance of BP and MF, we generated networks from the "four groups test" of [16] with a large number of variables $N$, four groups $q = 4$, average degree $c = p_0/N$, and ratio $\epsilon$ between the probability of being connected to a different group and within the same group. In other words, $\epsilon = c_{\rm out}/c_{\rm in}$. See an example adjacency matrix in Fig. 1. The results of inference using BP and MF are plotted in Fig. 2. From Fig. 2 we see several important points in which BP is superior over MF

- BP estimate gives better agreement with the true assignment. In the left and right part of Fig. 2 we see the following. In a region of large overlap, the two methods give the same result. This can be understood from the form of the BP and MF equations that become equivalent for very polarized marginals $\psi_r^i$. In the region of very small overlap both approaches converge to a fixed point that does not contain any information about the original group assignment. However, for parameter values close to the possible-impossible-inference phase transition the BP method gives larger overlap with the true group assignment than MF.
- BP is not over-confident. In the left and right part of Fig. 2 we compare the true overlap to the confidence value (21). For BP the two agree, just as they should if the marginals were evaluated exactly. In the MF approach, however, the confidence is considerably larger than the true overlap. This means that in the whole region where $C - Q > 0$, MF is misleadingly confident about the quality of the fixed point it found. The width of this region depends on the parameter values, but we observed that a good rule of thumb is that if the overlap reached is not very close to 1, then the MF method is unreliable.
- BP is faster. As we explained when we exposed the BP and MF equations, one iteration takes a comparable time for both methods. In the middle part of Fig. 2 we plot the number of iterations needed for convergence, we see that again around the phase transitions region MF needs more iterations to converge, and hence is overall slower that BP.
- BP does not converge to several different fixed points. Starting with randomly initialized messages, BP converged to the same fixed point (up to small fluctuations) in all the runs we observed. On the other hand in the region where

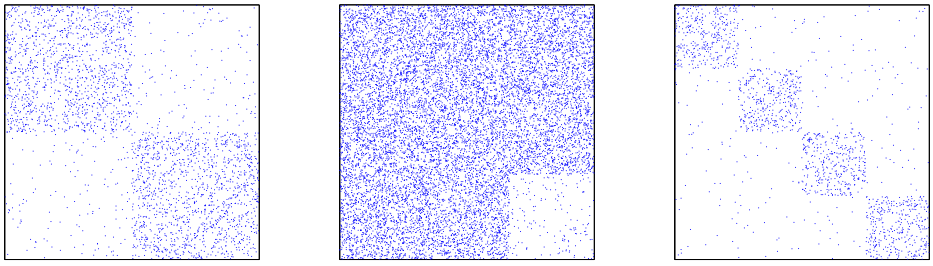


**Figure 1.** Adjacency matrices representing the block structure used for generating the various examples of the block model eq. (1) in this contribution. Rows and columns are ordered such that rows/columns corresponding to nodes with the same $t_i$ are next to each other. From left to right: a $q = 2$ modular network, a core-periphery structure, and a $q = 4$ modular network.

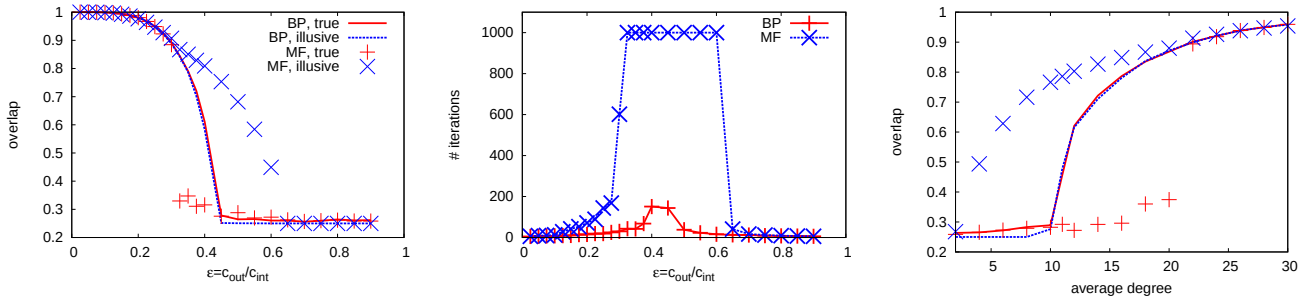



**Figure 2.** Comparison between the naïve mean field (MF) and belief propagation (BP) approaches to the E-step of expectation maximization. All datapoints correspond to networks with $N = 10^4$ nodes. The networks were generated using $q = 4$ groups, modular structure as sketched in left part of Fig. 1, and $c_{rr} = c_{\text{in}} > c_{rs} = c_{\text{out}}\ \forall s \neq r$. **Left:** True and illusive overlap $Q$ and $C$ for inference of the group assignment at different values of $\epsilon = c_{\text{out}}/c_{\text{in}}$. Note the transition between a phase where inference of class membership is possible and where it is not at $\epsilon_c = 0.43$. Also note that MF is overfitting the data, showing large illusive overlap $C$ in the region where inference is in fact impossible. **Middle:** The number of iterations needed for convergence of the E-step for the problem instances from the left part (we set the maximum number of iterations to be 1000). The computational effort is maximal at around $\epsilon_c$ for both methods, but BP converges faster. **Right:** True and illusive overlap $Q$ and $C$ at different values of the average connectivity $c = \langle k \rangle$ at fixed $\epsilon = 0.35$. Again, we observe a transition between feasible and infeasible inference at $\langle k \rangle_c(\epsilon)$ and the over-confidence of MF in the infeasible region.

the MF value of confidence $C$ differs from the true overlap $Q$ MF converged to several different fixed points depending on the initial conditions.

To summarize, BP for block model inference is superior to MF in terms of speed, of quality of the result and does not suffer from over-confidence the way MF does. Note that similar conclusions about BP compared to MF were reached for other inference problems in e.g. [24, 22].

An important point is that so far, have have discussed the situation of BP and MF algorithms using the known and correct values of parameters $p_r$, $p_{rs}$ in the E-step of expectation maximization. Concerning the M-step, we observed without surprise that the expectation maximization with BP gives better results than with MF in the region of parameters where BP is superior for the E-step. Otherwise the performance was comparable. Notably, both the approaches suffer from a strong dependence on the initial conditions of the parameters $p_{rs}^{t=0}$. This is a known problem in general expectation maximization algorithms [10]. The problem comes from the fact that the log-likelihood $\mathcal{L}(\theta)$ has many local maxima (each corresponding to a fixed point) in $\theta$ in which the expectation maximization update gets stuck. Fortunately the free energy serves as an indicator of which fixed point of EM is better. Hence a solution is to run the EM algorithm from many different initial conditions and to consider the fixed point with the smallest free energy (i.e. largest likelihood). Since the volume of possible parameters does not grow in the system size $N$, this still leads to an algorithm linear in the system size (for sparse networks). However, the increase in the running time is considerable and smarter initializations of the parameters $p_{rs}^{t=0}$ are desired. We introduce one such in the next section.

## 5.2 Spectral methods

Methods based on the eigenvectors of the adjacency matrix of the network provide one of the most flexible approaches of graph clustering problems applied in the practice and hence we compare the BP algorithm to this approach as well. The comparison of BP with modularity matrix based and random walker based spectral methods gives the following conclusions:

- In the case when the parameters $\theta$ are known and we search for the best estimate of the original group assignment we observed that BP is always better than the two spectral clustering algorithms (that is the random walker based and

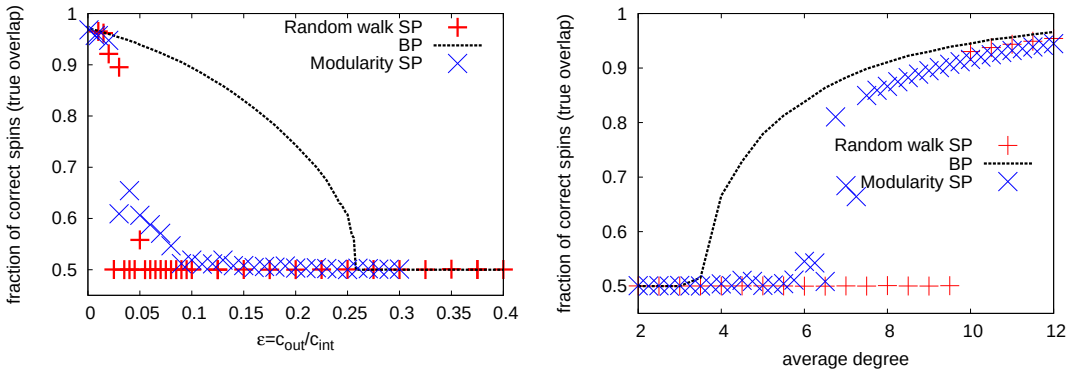



**Figure 3.** Comparison of the BP (only the E-step) and spectral clustering based on the random walker approach and the modularity matrix (see text). Datapoints correspond to networks with $N = 10^6$ nodes for the two spectral approaches and $N = 10^5$ for BP, and average degree $c = \langle k \rangle = 3$. The networks of $q = 2$ groups are generated using modular structure as sketched in the left part of Fig. 1. To ensure the random walk based method to work, we extracted the largest connected component of the network and ran the algorithm on it. **Left:** The overlap $Q$ at different values of $\epsilon = c_{\rm out}/c_{\rm in}$. Note how the spectral approaches can only correctly recover the latent class labels deep in the feasible region of the parameter space. **Right:** The overlap $Q$ at different values of the connectivity $c$ at fixed $\epsilon = 0.3$. Again, the spectral methods can only identify the latent class labels for problem instances well within the feasible region and fail on the hard instances near the critical connectivity.

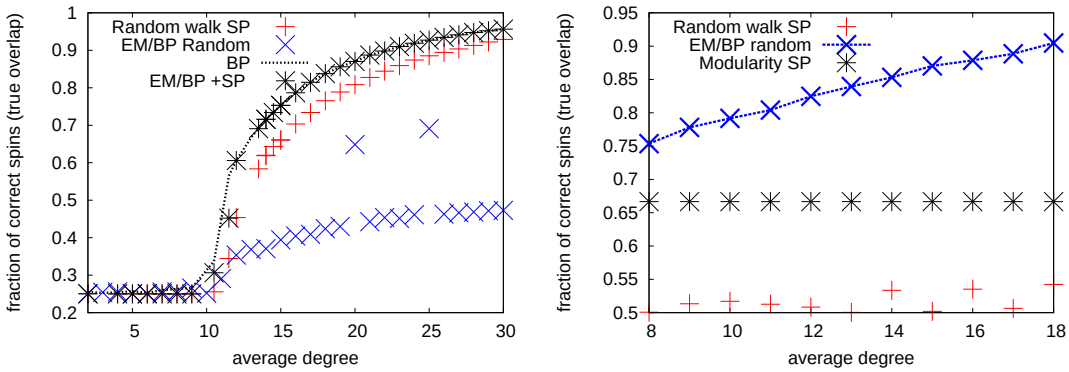



**Figure 4.** **Left:** An example where the EM with BP when initialized in a random matrix $c_{ab}$ does not work, whereas the random walker spectral method works well. The result of the spectral method serves as a initialization of $c_{ab}$ in the EM BP, which then improves the achieved overlap. Modular network of size $N = 10^5$ generated with $q = 4$ groups and $\epsilon = 0.35$. **Right:** An example where EM with BP works well even from random initial condition for the matrix $c_{ab}$, while spectral methods do not work well at all. The network exhibits a core periphery structure (middle panel of Fig.1) of size $N = 10^4$. Here average degree of core variables is equal to average degree of periphery variables. There are two groups of sizes $p_a = 2/3$ and $p_b = 1/3$, $c_{ab}$ matrix is in form of $\{c_{\rm in}, c_{io}; c_{io}, c_{\rm out}\}$, with $c_{\rm in} = \frac{9c}{8-\epsilon}$, $c_{\rm out} = \epsilon c_{\rm in}$ and $c_{io} = 1 - 0.5\epsilon$. The modularity based method gives overlap 2/3, because all variables were assigned to group 1.

the modularity based one) that we tested. This is illustrated in Fig. 3 and 4. In some cases (e.g. Fig. 4 left) the improvement BP provides over spectral methods is marginal. In other cases, e.g. for the core-periphery network of Fig. 4 right the improvement is drastic.

- A particularly important point we want to make is the following: For the cases tested in this paper the spectral methods are clearly suboptimal: there are regions where the BP inference gives large overlap while spectral clustering methods do not do better than chance. See for instance Fig. 3 left for $0.1 < \epsilon < 0.268$. Recently authors of [13] claimed "No other method will succeed in the regime where the modularity method fails", it was mentioned that their results may not be valid for networks with small average degree. Here we clearly show that for networks with small average degree the spectral methods are indeed not optimal. In our opinion, the conclusions of [13] apply only when the average degree diverges with the system size.
- A final point is that the spectral method should thus not be thought as the end of the story, but rather as the beginning: Indeed, they are extremely useful as a starting point for initializing EM BP to achieve improved overlap. This is shown in Fig. 4 left where EM BP starts from parameters taken from the result of the random walker based spectral method. This clearly improves the quality of the inference without having to restart EM BP for many initial conditions.

## 6 Conclusions

Using the example of latent variable inference in the stochastic block model of complex networks, we have compared belief propagation based inference techniques with traditional mean field approaches and classic spectral heuristics. To this end, we have used the recent discovery of a sharp transition in the parameter space of the stochastic block model from a phase where inference is possible to a phase where inference is provably impossible. In the vicinity of the phase transition, we find particularly hard problem instances that allow a performance comparison in a very controlled environment.

We could show that though spectral heuristics are appealing at first for their speed and uniqueness of the resulting decompositions, they only work reliably deep within the parameter region of feasible inference. In particular, very sparse graphs are difficult for spectral methods, as are block structures that are more complicated than a mere collection of cohesive subgraphs or communities. In short, they serve as a "quick and dirty" approach. We also evaluate if recent claims on the optimality of spectral methods for block structure detection hold for networks with small average degree [13].

Comparing naïve mean field techniques with belief propagation techniques, we find that the computational burden, which has so far hindered the wide spread use of belief propagation in fully connected graphical models such as block structure inference of (sparse or dense) networks, has been lifted completely. Not only is the computational complexity of the variable updates the same, belief propagation also exhibits much better convergence properties and this in particular on the hard problem instances. Hence, we expect that the presented formulations of belief propagation equations may find a wide range of application also in other fields of inference with fully connected graphical models. Note that the regime of $p_{rs} = O(1/N)$ considered here corresponds to the maximally sparse case. BP will still outperform other methods when $p_{rs} = O(N^{-\alpha})$ with $\alpha < 1$, albeit the performance differences will be much smaller.

Finally, we could show that using spectral decompositions in order to select initial conditions for learning the parameters of the stochastic block model can be a viable step in order to reduce the dependency on initial conditions when used in conjunction with expectation maximization type algorithms.

## Acknowledgments

We wish to thank to Cris Moore for discussions about various aspects of the EM BP algorithm. This work was supported by the Projet DIM "problématique transversales aux systèmes complexes" of the Institut des Systèmes Complexes, Paris Ìle-de-France (ISC-PIF). J.R. was supported by a Fellowship Computational Sciences of the Volkswagen Foundation.

## REFERENCES

[1] J. J. Daudin, F. Picard, and S. Robin, 'A mixture model for random graphs', *Statistics and Computing*, **18**(2), 173–183, (June 2008).

[2] A. Decelle, F. Krzakala, C. Moore, and L. Zdeborová, 'Asymptotic analysis of the stochastic block model for modular networks and its algorithmic applications', *Phys. Rev. E*, **84**, 066106, (Dec 2011).

[3] A. Decelle, F. Krzakala, C. Moore, and L. Zdeborová, 'Inference and phase transitions in the detection of modules in sparse networks', *Phys. Rev. Lett.*, **107**, 065701, (Aug 2011).

[4] A. P. Dempster, N. M. Laird, and D. B. Rubin, 'Maximum Likelihood from Incomplete Data via the EM Algorithm', *Journal of the Royal Statistical Society. Series B (Methodological)*, **39**(1), (1977).

[5] D. Easley and J. Kleinberg, *Networks, Crowds and Markets: Reasoning about a highly connected world*, Cambridge University Press, 2010.

[6] A. Goldenberg, A. X. Zheng, S.E. Fienberg, and E.M. Airoldi, 'A Survey of Statistical Network Models', *Foundations and Trends in Machine Learning*, **2**, 1–117, (2009).

[7] M. B. Hastings, 'Community detection as an inference problem', *Phys. Rev. E*, **74**, 035102, (2006).

[8] J. M. Hofman and C. H. Wiggins, 'Bayesian approach to network modularity', *Phys. Rev. Lett.*, **100**, 258701, (2008).

[9] P. W. Holland, K. B. Laskey, and S. Leinhard, 'Stochastic Block-models: First Steps', *Soc. Networks*, **5**, 109–137, (1983).

[10] D. Karlis and E. Xekalaki, 'Choosing initial values for the EM algorithm for fiite mixtures', *Computational Statistics and Data Analysis*, **41**, 577-590, (2003).

[11] S. Lafon and A. B. Lee, 'Diffusion Maps and Coarse-Graining: A Unified Framework for Dimensionality Reduction, Graph Partitioning, and Data Set Parameterization', *IEEE Trans. Patt. Ana. Mach. Intel.*, **28**(9), 1393, (2006).

[12] E. Mossel, J. Neeman, and A. Sly, 'Stochastic block models and reconstruction', *arxiv*, **1202.1499v3**, (2012).

[13] R. R. Nadakuditi and M. E. J. Newman, 'Graph spectra and the detectability of community structure in networks', *Phys. Rev. Lett.*, **108**, 188701, (May 2012).

[14] R. Neal and G. E. Hinton, 'A view of the em algorithm that justifies incremental, sparse, and other variants', in *Learning in Graphical Models*, pp. 355–368. Kluwer Academic Publishers, (1998).

[15] M. E. J. Newman, 'Finding community structure in networks using the eigenvectors of matrices', *Phys. Rev. E*, **74**, 036104, (Sep 2006).

[16] M. E. J. Newman and M. Girvan, 'Finding and evaluating community structure in networks', *Phys. Rev. E*, **69**, 026113, (Feb 2004).

[17] M. E. J. Newman, *Complex Networks: An Introduction*, Oxford University Press, 2010.

[18] M. Onsjö and W. Watanabe, *A Simple Message Passing Algorithm for Graph Partitioning Problems*, 507–516, number 4228 in LNCS, Springer Berlin / Heidelberg, 2006.

[19] S. Pinkert, J. Schultz, and J. Reichardt, 'Protein-Interaction Networks-More than mere Modules', *PLoS Comput Biolo*, **6**(1), e1000659, (2010).

[20] J. Reichardt and M. Leone, '(Un)detectable cluster structure in sparse networks', *Phys Rev Lett*, **101**, 078701, (2008).

[21] J. Reichardt, R. Alamino, and D. Saad, 'The interplay between microscopic and mesoscopic structures in complex networks', *PLoS ONE*, **6**(8), e21282, (08 2011).

[22] P. Sen and L. Getoor, 'Link-based classification', Technical report, Univresity of Maryland Technical Reports CS-TR-4858, (2007).

[23] P. Uetz, L. Giot, G. Gagney, T.A. Mansfield, R.S. Judson, J.R. Knight, D. Lockshon, V. Narayan, M. Srinivasan, P. Pochart, A. Querschi-Emili, Y. Li, B. Goldwin, D. Conover, T. Kalbfleisch, G. Vijayadamodar, M. Yang, M. Johnston, S. Fields, and J.M. Rothberg, 'A comprehensive analysis of protein-protein interactions in saccharomyces cerevisiae', *Nature*, **403**(6770), 623–7, (2000).

[24] Y. Weiss, *Advanced Mean Field Methods: Theory and Practice*, chapter 15, 229, MIT Press, 2001.

[25] B. Wellman, 'Computer Networks as Social Networks', *Science*, **293**(5537), 2031–2034, (2001).

[26] J. S. Yedidia, W. T. Freeman, and Y. Weiss, 'Understanding belief propagation and its generalizations', in *International Joint Conference on Artificial Intelligence (IJCAI)*, (2001).