http://pubsonline.informs.org/journal/ijoo

INFORMS JOURNAL ON OPTIMIZATION
Vol. 2, No. 3, Summer 2020, pp. 209–228
ISSN 2575-1484 (print), ISSN 2575-1492 (online)

Integer Programming on the Junction Tree Polytope for Influence Diagrams

Axel Parmentier,[a] Victor Cohen,[a] Vincent Leclère,[a] Guillaume Obozinski,[b] Joseph Salmon[c]

[a] CERMICS, Ecole des Ponts, 77455 Marne-la-Vallée, France; [b] Swiss Data Science Center, EPFL and ETH Zürich, 8006 Zürich, Switzerland; [c] IMAG, Université de Montpellier, CNRS, 34090 Montpellier, France

Contact: axel.parmentier@enpc.fr, https://orcid.org/0000-0003-1762-4947 (AP); victor.cohen@enpc.fr, https://orcid.org/0000-0002-2616-8800 (VC); vincent.leclere@enpc.fr, https://orcid.org/0000-0002-5757-6655 (VL); guillaume.obozinski@epfl.ch, https://orcid.org/0000-0002-7629-7571 (GO); joseph.salmon@umontpellier.fr, https://orcid.org/0000-0002-3181-0634 (JS)

Received: July 30, 2019
Revised: January 15, 2020; April 22, 2020
Accepted: May 3, 2020
Published Online in Articles in Advance: July 24, 2020

https://doi.org/10.1287/ijoo.2019.0036

Abstract. Influence diagrams (ID) and limited memory influence diagrams (LIMID) are flexible tools to represent discrete stochastic optimization problems, with the Markov decision process (MDP) and partially observable MDP as standard examples. More precisely, given random variables considered as vertices of an acyclic digraph, a probabilistic graphical model defines a joint distribution via the conditional distributions of vertices given their parents. In an ID, the random variables are represented by a probabilistic graphical model whose vertices are partitioned into three types: chance, decision, and utility vertices. The user chooses the distribution of the decision vertices conditionally to their parents in order to maximize the expected utility. Leveraging the notion of rooted junction tree, we present a mixed integer linear formulation for solving an ID, as well as valid inequalities, which lead to a computationally efficient algorithm. We also show that the linear relaxation yields an optimal integer solution for instances that can be solved by the "single policy update," the default algorithm for addressing IDs.

Funding: This work was partially funded by the Operations Research and Machine Learning between Air France and Ecole des Ponts.
Supplemental Material: The online appendix is available at https://doi.org/10.1287/ijoo.2019.0036.

Keywords: influence diagrams • partially observed Markov decision processes • probabilistic graphical models • linear programming

1. Introduction

In this paper we seek to address stochastic optimization problems with structured information and discrete decision variables via mixed integer linear reformulations. We start by recalling the framework of influence diagrams (IDs; more details can be found in Koller and Friedman 2009, chapter 23) and present the classical linear formulation for some special cases.

1.1. The Framework of Parameterized ID

Let $G = (V, E)$ be an acyclic directed graph, and for each vertex v in V, let X_v be a random variable taking a value in a finite state space $\mathcal{X}_v$. For any $C \subset V$, let X_C denote $(X_v)_{v \in C}$, and let $\mathcal{X}_C$ be the Cartesian product $\mathcal{X}_C = \prod_{v \in C} \mathcal{X}_v$. We say that the distribution of the random vector X_V factorizes as a *directed graphical model* on G if, for all $x_V \in \mathcal{X}_V$, we have

$$\mathbb{P}(X_V = x_V) = \prod_{v \in V} p_{v|\mathrm{pa}(v)}(x_v | x_{\mathrm{pa}(v)}), \tag{1}$$

where $\mathrm{pa}(v)$ is the set of parents of v, that is, the set of vertices u such that (u, v) belongs to E, and $p_{v|\mathrm{pa}(v)}(x_v | x_{\mathrm{pa}(v)}) = \mathbb{P}(X_v = x_v | X_{\mathrm{pa}(v)} = x_{\mathrm{pa}(v)})$. Further, given an arbitrary collection of conditional distributions $\{p_{v|\mathrm{pa}(v)}\}_{v \in V}$, Equation (1) uniquely defines a probability distribution on $\mathcal{X}_V$.

Let $(V^{\mathrm{a}}, V^{\mathrm{c}}, V^{\mathrm{r}})$ be a partition of V, where V^{c} is the set of *chance vertices*, V^{a} is the set of *decision vertices*, and V^{r} is the set of *utility vertices* (the ones with no descendants). For ease of notation we denote $V^{\mathrm{s}} = V^{\mathrm{c}} \cup V^{\mathrm{r}}$. Letters a, r, and s, respectively, stand for action, reward, and stochastic in V^{a}, V^{r}, and V^{s}. We say that $G = (V^{\mathrm{s}}, V^{\mathrm{a}}, E)$ is an *ID*. Consider a set of conditional distributions $\mathfrak{p} = \{p_{v|\mathrm{pa}(v)}\}_{v \in V^{\mathrm{c}} \cup V^{\mathrm{r}}}$ and a collection of *reward functions* $r = \{r_v\}_{v \in V^{\mathrm{r}}}$ with $r_v : \mathcal{X}_v \to \mathbb{R}$. Then we call $(G, \mathcal{X}_V, \mathfrak{p}, r)$ a *parameterized influence diagram* (PID).[1] We will sometimes refer to the parameters $(\mathcal{X}_V, \mathfrak{p}, r)$ by ρ for conciseness.

Let Δ_v denote the set of conditional distributions $\delta_{v|\mathrm{pa}(v)}$ on $\mathcal{X}_v$ given $\mathcal{X}_{\mathrm{pa}(v)}$. Given the set of conditional distributions $\mathfrak{p}$, a *strategy* δ in $\Delta = \prod_{v \in V^{\mathrm{a}}} \Delta_v$ uniquely defines a distribution $\mathbb{P}_\delta$ on $\mathcal{X}_V$ through

$$\mathbb{P}_\delta(X_V = x_V) = \prod_{v \in V^{\mathrm{s}}} p_{v|\mathrm{pa}(v)}\big(x_v | x_{\mathrm{pa}(v)}\big) \prod_{v \in V^{\mathrm{a}}} \delta_{v|\mathrm{pa}(v)}\big(x_v | x_{\mathrm{pa}(v)}\big). \tag{2}$$

Let $\mathbb{E}_\delta$ denote the corresponding expectation. The *maximum expected utility* (MEU) problem associated to the PID $(G, \mathcal{X}_V, \mathfrak{p}, r)$ is the maximization problem

$$\max_{\delta \in \Delta} \quad \mathbb{E}_\delta\left(\sum_{v \in V^{\mathrm{r}}} r_v(X_v)\right). \tag{3}$$

A strategy $\delta \in \Delta^{\mathrm{d}} \subset \Delta$ is *deterministic* if, for every $v \in V^{\mathrm{a}}$ and any $x_v, x_{\mathrm{pa}(v)} \in \mathcal{X}_v \times \mathcal{X}_{\mathrm{pa}(v)}$, $\delta_{v|\mathrm{pa}(v)}(x_v|x_{\mathrm{pa}(v)})$ is a Dirac measure. It is well known that there always exists an optimal solution to MEU (3) that is deterministic (see, e.g., Liu 2014, lemma C.1 for a proof).

Remark 1. A common practice in the literature is to define utility vertices $v \in V^{\mathrm{r}}$ that are deterministic functions $f(x_{\mathrm{pa}(v)})$ of their parents. This case can of course be modeled in our setting: For each v in V^{r}, it suffices to define state spaces $\mathcal{X}_v = \mathcal{X}_{\mathrm{pa}(v)}$, conditional probabilities $p_{v|\mathrm{pa}(v)}(x_v|x_{\mathrm{pa}(v)})$ to be equal to 1 if $x_v = x_{\mathrm{pa}(v)}$ and 0 otherwise, and reward functions $r(x_v) := f(x_v)$.

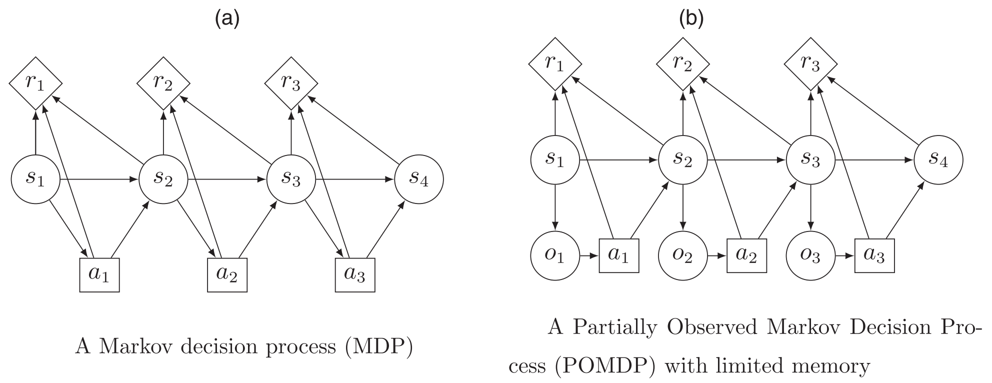
Remark 2. In an ID, one says that there is *perfect recall* of previous actions, when, for a given topological order, the sets of parents of a given action are all the actions that appear before it in the topological order and their parents. In the absence of *perfect recall*, many authors have used the expression *limited memory influence diagram* (LIMID) to characterize the corresponding ID (e.g., Lauritzen and Nilsson 2001). In this paper, we consider the general case of LIMIDs but we refer to them as *IDs* throughout the paper, following the convention adopted in Koller and Friedman (2009, chapter 23).

We conclude this section with some classical examples of IDs, shown in Figure 1.

Example 1. Consider a maintenance problem in which at time t a machine is in state s_t. The action a_t taken by the decision maker according to the current state is typically a binary decision which is to either perform maintenance on it (which is costly) or not (which increases the probability of failure). The problem is considered over a finite horizon with multiple maintenance interventions possible. State and decision together lead to a new (random) state s_{t+1}, and the triple (s_t, a_t, s_{t+1}) induces a reward r_t. This is an example of a *Markov decision process* (MDP) which is probably the simplest type of ID, represented in Figure 1(a).

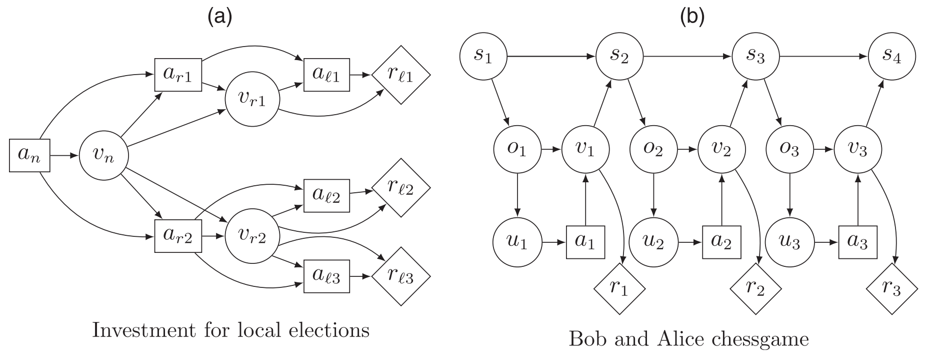
In practice, the actual state s_t of the machine is often not known, and we have instead an observation o_t that only carries partial information about the state, which leads to a more complex ID known as a *partially observed Markov decision process* (POMDP). Taking decisions based on all past observations and decisions (which is the *perfect recall* case) would lead to a better MEU, but requires working with policies living in spaces of exponentially large dimension and thus leads to intractable MEU problems (Papadimitriou and Tsitsiklis 1987). To reduce the complexity of the MEU problem, we follow Lauritzen and Nilsson (2001) and restrict ourselves to *memoryless policies*: as illustrated in Figure 1(b), the decision a_t is taken based on observation o_t.

Figure 1. ID Examples


(a)
r1
r2
r3
s1
s2
s3
s4
a1
a2
a3
A Markov decision process (MDP)
(b)
o1
o2
o3
A Partially Observed Markov Decision Process (POMDP) with limited memory

Note. We represent chance vertices (V^{s}) in circles, decision vertices (V^{a}) in rectangles, and utility vertices (V^{r}) in diamonds.

Figure 2. IDs of Examples 2 and 3



Example 2. Figure 2(a) depicts an ID modeling the media investment strategy of a political party for the next elections. The national committee starts in a_n by deciding how much to invest into national media coverage and which budget it gives to regional committees. Based on the national popularity rating v_n after the interventions on national media, a regional committee i decides which fraction a_{ri} of their funds they allocate to regional media and local committees. Based on regional popularity rating v_{ri} after the interventions on regional media, each local committee j decides how much to invest in local meetings and local media $a_{\ell j}$. The objective maximized is the total number of local elections $r_{\ell j}$ won.

Example 3. Consider two chess players: Bob and Alice. They play chess, and for each game they bet a symbolic coin. However, they can refuse to play. Suppose that Alice wants to play chess every day.[2] On day t, she has a current confidence level s_t. On the day of the game, her current mental fitness is denoted o_t. When Bob meets with Alice, he makes the decision to play depending on her demeanor, denoted u_t. Then Bob can accept or decline the challenge, and his decision is denoted a_t. Let v_t denote the winner (getting a reward r_t). If Bob declines the challenge, there is no winner and no reward. Then, Alice's next confidence level is affected by the result of the game and her previous confidence level. This stochastic decision problem can be modeled by an ID as shown in Figure 2(b).

1.2. Solving MDP Through Linear Programs

We review here a well-known linear programming formulation for MDPs (see d'Epenoux 1963), which is a special case of the mixed integer linear program (MILP) formulation that we will introduce subsequently in the paper. We denote by $p(s'|s,a)$ the probability of transitioning from state s to state s' if action a is taken, and we denote by $r(s,a,s')$ the reward associated to this transition. For $t \in \{1,\dots,T\}$, let μ_s^t represent the probability of being in state s at time t, let μ_{sa}^t represent the probability of being in state s and taking action a at time t, and let $\mu_{sas'}^t$ represent the probability of being in state s and taking action a at time t, while transiting to state s' at time $t+1$. This leads to the linear program

$$\max_{\mu} \quad \sum_{t=1}^{T-1} \sum_{s,a,s'} \mu_{sas'}^t \, r(s,a,s') \tag{4a}$$

$$\text{s.t.} \quad \mu_{sas'}^t = p(s'|s,a)\,\mu_{sa}^t, \quad \forall t,s,s', \tag{4b}$$

$$\mu_{s_0}^0 = 1, \tag{4c}$$

$$\sum_{s,a} \mu_{sas'}^t = \mu_{s'}^{t+1}, \forall t,s', \quad \sum_{s'} \mu_{sas'}^t = \mu_{sa}^t, \forall t,s,a, \ \text{ and } \ \sum_a \mu_{sa}^t = \mu_s^t, \quad \forall t,s, \tag{4d}$$

$$\sum_s \mu_s^t = 1, \quad \forall t, \tag{4e}$$

$$\mu_s^t, \mu_{sa}^t, \mu_{sas'}^t \in [0,1], \quad \forall t,a,s,s', \tag{4f}$$

where the objective (4a) is simply the expected reward, the constraints (4b) represent the state dynamics, the constraints (4c) set the initial state of the system to s_0, and the constraints (4d)–(4f) ensure that μ represents the marginals laws of a joint distribution.

1.3. Literature

IDs were introduced by Howard and Matheson (1984) (see also Howard and Matheson 2005) to model stochastic optimization problems using a probabilistic graphical model framework. Originally, the decision makers were assumed to have *perfect recall* (Shachter 1986, Shenoy 1992, Jensen et al. 1994) of the past actions.

Lauritzen and Nilsson (2001) relaxed this assumption and provided a simple (coordinate descent) algorithm to find a good strategy: the single policy update (SPU) algorithm. These authors used the name *limited memory IDs* when relaxing the perfect recall assumption, but we follow the convention of Koller and Friedman (2009) who still call them IDs. The same authors also introduced the notion of *soluble* ID as a sufficient condition for SPU to converge to an optimal solution. This notion has been generalized by Koller and Milch (2003) to obtain a necessary and sufficient condition. In general, SPU only finds a locally optimal strategy and requires an exact inference, so that it is therefore limited by the *treewidth* (Chandrasekaran et al. 2008). More recently, Mauá and de Campos (2011) and Mauá and Cozman (2016) have introduced a new algorithm, *multiple policy update*, which has both an exact and a heuristic version and relies on a concept of dominance to discard partial solutions. It can be interpreted as a generalization of SPU where several decisions are considered simultaneously. Later on, Khaled et al. (2013) proposed a similar approach, in the spirit of branch and bound, whereas Liu (2014) introduced heuristics based on approximate variational inference. Usually, inference computations in IDs are done within valuation algebra on the pair potential-utility (Jensen et al. 1994, Dechter 2013). Lee et al. (2018) propose an inference algorithm providing upper bounds on the MEU which uses valuation algebras for IDs (Dechter 2013). We choose to use the *marginal polytope* for inference computations, because it is useful for mathematical programming approaches (Wainwright and Jordan 2008, Sontag et al. 2011).

Finally, the problem of solving an ID can be polynomially transformed into a maximum a posteriori (MAP) problem and, hence, can be solved using popular MAP solvers such as toulbar2 (Hurley et al. 2016). For further details about the transformation, see Cano et al. (1994), Antonucci and Zaffalon (2008) and Maua (2016).

Finding an optimal strategy for an ID has been shown to be NP hard even when restricted to IDs of treewidth no greater than two or to trees with binary variables (Mauá et al. 2012a, 2013). Note that even obtaining an approximate solution is also NP hard (Mauá et al. 2012a).

Beyond the classical linear programming formulation for MDPs, mathematical programming formulations have been proposed for some special cases of IDs, including decomposable or weakly coupled MDPs (Bertsimas and Niño-Mora 2000, De Farias and Van Roy 2003, Hawkins 2003, Adelman and Mersereau 2008, Bertsimas and Mišić 2016) and POMDP with perfect recall and short horizon (Aras and Dutech 2010). Among these formulations, the one of Bertsimas and Mišić (2016) is the closest to ours, since it also relies on variables that correspond to moments or distributions. The variables of the other formulations correspond to time averages (Bertsimas and Niño-Mora 2000) or value functions (De Farias and Van Roy 2003, Hawkins 2003, Adelman and Mersereau 2008), which makes these formulations harder to generalize to IDs.

Credal networks are generalizations of probabilistic graphical models where the parameters of the model are not known exactly. MILP formulations for credal networks that could be applied to IDs have been introduced by de Campos and Cozman (2007) and de Campos and Ji (2012). However, the number of variables they require is exponential in the *pathwidth*, which can be arbitrarily larger than the width of the tree we are using (this follows from Scheffler (1990, theorem 4)), and the linear relaxation of their MILP is not as good as the one of the MILP we propose and does not yield an integer solution on soluble IDs. Our approach can naturally be extended to credal networks.

Finally, Examples 1, 2, and 3 are sequential decisions problems in stochastic optimization. Many different solution approaches have been proposed under different names in different academic communities. Although describing these approaches is beyond the scope of this paper, we refer the interested reader to the tutorial of Powell (2014). As we have already mentioned in Example 1, we consider only memoryless policies for POMDP (Littman 1994a). Li et al. (2011) emphasize the benefits of using memoryless policies in practice. In the literature, POMDPs are generally considered with the perfect recall assumption. Exact approaches to that case generally transform a POMDP into an equivalent MDP on the *belief state* space (Eckles 1968) and solve that MDP by dynamic programming (Smallwood and Sondik 1973, Littman 1994b). However, these exact algorithms become quickly intractable when the size of the spaces grows, and many heuristics have been proposed (Ross et al. 2008, Shani et al. 2013).

1.4. Contributions

The contributions of the paper are as follows.

- We introduce a nonlinear program (NLP) and a MILP for the MEU problem on IDs.
- These mathematical programs rely on a variant of the concept of a *strong junction tree* which we introduce and call a *rooted junction tree* (RJT). We provide algorithms to build RJTs that lead to "good" mathematical programs for IDs.
- We introduce a particular form of valid cuts for the obtained MILP. These valid cuts leverage conditional independence properties in the ID. We show that our cuts are the strongest ones in a certain sense. We believe

that this idea of leveraging conditional independence to obtain valid cuts is fairly general and could be extended to other contexts.

• We establish a link between the linear relaxation of our MILP and the concept of *soluble relaxation* previously introduced in the literature on IDs. In fact, our relaxation provides a better bound than those relaxations.

• We provide two new characterizations of soluble IDs: first, as the only IDs that can be solved to optimality using the linear relaxation of our MILP; and second, and more importantly, as the IDs for which there exists a junction tree such that the set of collections of moments of distributions that are induced by the different policies is convex.

• We illustrate our mathematical programs and their properties on some simple numerical examples.

Our approach obviously has limitations. Indeed, any exact method to solve Problem (3) must compute the exact value of $\mathbb{E}_\delta(\sum_{v\in V^r} r_v(X_v))$ when it evaluates a strategy δ. Since exact inference is exponential in the treewidth, this type of method is limited in practice to graphs with moderate treewidth. Our approach to solving (3) relies on the *RJTs* that we introduce and is therefore practically limited to IDs with moderate *rooted treewidth*. This is an additional limitation since the rooted treewidth can be significantly larger than the usual treewidth. Actually, on pathological cases such as the one in Figure 11 at the end of Appendix A, the rooted treewidth can be even worse than the *pathwidth* (Robertson and Seymour 1983). We however expect our method to work well on two important class of applications, for which the rooted treewidth is of the same order of magnitude as the treewidth: temporal models like those considered in Examples 1 and 3, and strategic decision problems with several decision levels in large organizations as in Example 2. The code to perform numerical experiments is available here.[3]

1.5. Organization of the Paper

In Section 2, we recall some definitions for graphical models, which are used to extend the notion of *junction tree* to *RJT* in Section 3. With these tools, Section 4 introduces a bilinear formulation that can be rewritten as a MILP formulation to the MEU Problem (3). In Section 5 we propose efficient valid cuts for the MILP formulation and interpret them in terms of graph relaxations. Section 6 studies the polynomial case of soluble IDs, showing that the IDs that can be solved to optimality by SPU can be solved with (continuous) linear programming using our formulation. Finally, Section 7 summarizes our numerical experiments.

2. Background on Probabilistic Graphical Models

In this section, we recall notation and tools used in the following sections to refomulate the MEU Problem (3). Those are well-known results from probabilistic graphical model theory that do not yet pertain to IDs.

2.1. Graph Notation

This section introduces our notations for graphs, which are for the most part the ones commonly used in the combinatorial optimization community (Schrijver 2003). A directed graph G is a pair (V,E) where V is the set of vertices and $E \subseteq V^2$ is the set of arcs. We write $u \to v$ when $(u,v) \in E$. Let $[k] := \{1,\dots,k\}$. A *path* is a sequence of vertices $v_1,\dots,v_k$ such that $v_i \to v_{i+1}$ for any $i \in [k-1]$. A path between two vertices u and v is called a u-v path. We write $u \overset{G}{\dashrightarrow} v$ to denote the existence of a u-v path in G, or simply $u \dashrightarrow v$ when G is clear from context. We write $u \leftrightarrows v$ if there is an arc $u \to v$ or $v \to u$. A *trail* is a sequence of vertices $v_1,\dots,v_k$ such that $v_i \leftrightarrows v_{i+1}$, for all $i \in [k-1]$. A *parent* (resp., *child*) of a vertex v is a vertex u such that (u,v) (resp., (v,u)) belongs to E; we denote by $\mathrm{pa}(v)$ the set of parents vertices (resp., $\mathrm{ch}(v)$ the set of children vertices). The *family* of v, denoted by $\mathrm{fa}(v)$, is the set $\{v\} \cup \mathrm{pa}(v)$. A vertex u is an *ancestor* (resp., a *descendant*) of v if there exists a u-v path (resp., a v-u path). We denote, respectively, by $\mathrm{anc}(v)$ and $\mathrm{des}(v)$ the set of ancestors and descendants of v. Finally, let $\overline{\mathrm{anc}}(v) = \{v\} \cup \mathrm{anc}(v)$ and $\overline{\mathrm{des}}(v) = \{v\} \cup \mathrm{des}(v)$. For a set of vertices C, the parent set of C, again denoted by $\mathrm{pa}(C)$, is the set of vertices u that are parents of a vertex $v \in C$. We define similarly $\mathrm{fa}(C)$, $\mathrm{ch}(C)$, $\mathrm{anc}(C)$, $\overline{\mathrm{anc}}(C)$, $\mathrm{des}(C)$, and $\overline{\mathrm{des}}(C)$. Note that we sometimes indicate with a subscript the graph according to which the parents, children, and so on are taken. For instance, $\mathrm{pa}_G(v)$ denotes the parents of v in G.

A *cycle* is a path $v_1,\dots,v_k$ such that $v_1 = v_k$. A graph is *connected* if there exists a path between any pair of vertices. An *acyclic* graph is a graph which has no cycle. An undirected graph is a *tree* if it is connected and acyclic. A directed graph is a *directed tree* if its underlying undirected graph is a tree. A *rooted tree* is a directed tree such that all vertices have a common ancestor referred to as the *root* of the tree.[4] In a rooted tree, all vertices but the root have exactly one parent.

2.2. Directed Graphical Model

In this paper, given three random variables X, Y, and Z, the notation $X \perp\!\!\!\perp Y|Z$ stands for "X is independent from Y given Z."

A well-known sufficient condition for a distribution to factorize as a probabilistic graphical model is that each vertex is independent from its nondescendants given its parents.

Proposition 1 (Koller and Friedman 2009, theorem 3.1, p. 62). *Let X_V be a random variable on $\mathcal{X}_V$. Then its distribution is said to factorize according to G, that is,*

$$\mathbb{P}(X_V = x_V) = \prod_{v\in V} \mathbb{P}\big(X_v = x_v | X_{\mathrm{pa}(v)} = x_{\mathrm{pa}(v)}\big),$$

if and only if

$$X_v \perp\!\!\!\perp X_{V\backslash\overline{\mathrm{des}_G(v)}} | X_{\mathrm{pa}(v)} \quad \textit{for all } v \textit{ in } V. \tag{5}$$

Note that this result is sometimes considered as the counterpart of the theorem of Hammersley and Clifford (Koller and Friedman 2009, theorem 4.2, p. 116) but for directed graphical models.

2.3. Junction Trees

To solve the MEU Problem (3), one needs to consider distributions μ_V on $\mathcal{X}_V$ that factorize as in (2) for some strategy δ. In theory, it suffices to consider distributions μ_V satisfying the conditional independences given Equation (5) and such that $\mathbb{P}_\mu(X_v|X_{\mathrm{pa}(v)}) = p_{v|\mathrm{pa}(v)}$ for each vertex v that is not a decision. However, the joint distribution μ_V on all the variables is too large to be manipulated in practice as soon as V is moderately large. In that case, it is useful to work with a *vector of moments* $\tau = (\tau_C)_{C\in\mathcal{V}}$, where $\mathcal{V} \subseteq 2^V$, that is, a vector of distributions τ_C on subsets of variables C of tractable size. A vector of moments $(\tau_C)_{C\in\mathcal{V}}$ derives from a distribution μ_V on $\mathcal{X}_V$ if each moment $\tau_C \in [0,1]^{\mathcal{X}_C}$ is the marginal distribution induced by μ_V on C, that is, $\tau_C(x_C) = \sum_{x_{V\backslash C}\in\mathcal{X}_{V\backslash C}} \mu_V(x_C, x_{V\backslash C})$ for all C in $\mathcal{V}$ and x_C in $\mathcal{X}_C$. To keep notations light, we will write this type of equality more compactly as $\tau_C = \sum_{x_{V\backslash C}} \mu_V$. We use the notation $\mu = (\mu_C)_{C\in\mathcal{V}}$ for the vector of moments deriving from a distribution, and we use $\mathbb{P}_\mu$ or μ_V for the corresponding distribution on $\mathcal{X}_V$.

A necessary condition for a vector of moments $(\tau_C)_{C\in\mathcal{V}}$ to derive from a distribution is to be *locally consistent*, that is, to induce the same marginals on the intersections of pairs of elements of $\mathcal{V}$, that is, for all $C_1, C_2 \in \mathcal{V}$, we have

$$\sum_{x_{C_1\backslash C_2}} \tau_{C_1} = \sum_{x_{C_2\backslash C_1}} \tau_{C_2},$$

where, as before, $\sum_{x_{C_1\backslash C_2}} \tau_{C_1}$ is the vector $\big(\sum_{x_{C_1\backslash C_2}\in\mathcal{X}_{C_1\backslash C_2}} \tau_{C_1}(x_{C_1\backslash C_2}, x_{C_1\cap C_2})\big)_{x_{C_1\cap C_2}\in\mathcal{X}_{C_1\cap C_2}}$. It turns out that graphical model theory provides a condition on the choice of $\mathcal{V}$ together with the choice of local consistency constraints which are sufficient for $(\tau_C)_{C\in\mathcal{V}}$ to derive from a distribution on $\mathcal{X}_V$. This is done via the definition of a junction tree. Let $\mathcal{G} = (\mathcal{V}, \mathcal{A})$ be an undirected graph associated with $G = (V, E)$ with $\mathcal{V} \subseteq 2^V$. If $\mathcal{G}$ is a tree and satisfies the *running intersection property*, that is, given two vertices C_1 and C_2 in $\mathcal{V}$, any vertex C on the unique undirected path from C_1 to C_2 in $\mathcal{G}$ satisfies $C_1 \cap C_2 \subset C$, then $\mathcal{G}$ is called a *junction tree* of G. See Figure 4 for an illustration of this notion. Given a junction tree $\mathcal{G}$, its associated *marginal polytope* $\mathcal{M}^0_{\mathcal{G}}$ can be defined as

$$\mathcal{M}^0_{\mathcal{G}} = \left\{ (\tau_C)_{C\in\mathcal{V}} \left| \begin{array}{llll} \tau_C \geq 0 & \text{and} \quad \sum_{x_C} \tau_C(x_C) = 1, & & \forall x_C \in \mathcal{X}_C, \forall C \in \mathcal{V}, \\ \text{and} & \sum_{x_{C_1\backslash C_2}} \tau_{C_1} = \sum_{x_{C_2\backslash C_1}} \tau_{C_2}, & & \forall \{C_1, C_2\} \in \mathcal{A}, \end{array} \right. \right\} \tag{6}$$

Then $\tau = (\tau_C)_{C\in\mathcal{V}}$ is a vector of moments deriving from a distribution μ_V on $\mathcal{X}_V$ if and only if $\tau \in \mathcal{M}^0_{\mathcal{G}}$ (Wainwright and Jordan 2008, proposition 2.1). Moreover, if we introduce the set of *separators*[5] $\mathcal{S} = \{C_1 \cap C_2 \mid \{C_1, C_2\} \in \mathcal{A}\}$, then we have $\mu_V(x_V) = (\prod_{C\in\mathcal{V}} \tau_C(x_C))/(\prod_{S\in\mathcal{S}} \tau_S(x_S))$. The *width* of a junction tree $\mathcal{G} = (\mathcal{V}, \mathcal{A})$ on $G = (V, E)$ is the maximum number of vertices of V in a vertex C of $\mathcal{V}$ minus one. The *treewidth* of a graphical model G is the minimum width of a junction tree on G.

3. RJTs

To solve the MEU Problem (3), we work on vectors of moments $(\mu_C)_{C\in\mathcal{V}}$ that correspond to the moments of distributions μ induced by policies $\delta \in \Delta$. Hence, we are interested in vectors μ of moments such that the unique corresponding distribution μ_V on $\mathcal{X}_V$ factorizes as a directed graphical model on G. Such vectors of moments necessarily satisfy a "local" version of the conditional independence condition (5), namely,

$$\forall C \in \mathcal{V}, \qquad \Big((Z_C \sim \mu_C) \quad \Rightarrow \quad \Big(Z_v \perp\!\!\!\perp Z_{C\backslash\overline{\mathrm{des}(v)}} | Z_{\mathrm{pa}(v)},\ \forall v \in V : \mathrm{fa}(v) \subseteq C\Big)\Big). \tag{7}$$

Given a vector of moments μ in the marginal polytope of a junction tree $(\mathcal{V},\mathcal{A})$, there exists a unique distribution μ_V on $\mathcal{X}_V$ that factorizes according to the junction tree $\mathcal{G}$. However, for a generic junction tree, if $\mu = (\mu_C)_{C\in\mathcal{V}}$ is a vector of the moments of the marginal polytope, the fact that, for all $C \in \mathcal{V}$, property (7) holds is not a sufficient condition for μ to be the moments of a distribution μ_V that factorizes on G. For instance, on the junction tree of Figure 3(b), Equation (7) does not enforce the independence of u and v, which is required on the graph of Figure 3(a). But it becomes a sufficient condition under the additional assumption that $(\mathcal{V},\mathcal{A})$ is an "RJT," a notion that we introduce in this section and develop in more details in Appendix A.

3.1. Definition and Main Properties

From now on and in the rest of the paper, we consider a *rooted* tree $\mathcal{G} = (\mathcal{V},\mathcal{A})$ whose underlying undirected graph is a junction tree on a graph $G = (V,E)$. For any $v \in V$, by the running intersection property, the subgraph $\mathcal{G}_v$ of $\mathcal{G}$ induced on the nodes C of $\mathcal{V}$ containing v is a tree, and given that $\mathcal{G}$ is a rooted tree, so is $\mathcal{G}_v$. We call the root of $\mathcal{G}_v$ the *root clique of* v and denote it by C_v.

Definition 1. An *RJT* on $G = (V,E)$ is a rooted tree with nodes in 2^V such that

i. its underlying undirected graph $\mathcal{G} = (\mathcal{V},\mathcal{A})$ is a junction tree,
ii. for all $v \in V$, we have $\mathrm{fa}(v) \subseteq C_v$.

Let $\mathcal{G}$ be an RJT on G, and let v be a vertex of V. Given $C \in \mathcal{V}$, let the offspring of C be defined by $\mathrm{offspring}(C) = \{v \in V : C_v = C\}$, where C_v is the above-defined root clique of v, and let $\check{C}$ denote $C\backslash\mathrm{offspring}(C)$.

See Figure 4 for an example of this notion. Note that an RJT always exists. Indeed, the cluster graph composed of a single vertex $C = V$ is an RJT. Algorithms to build interesting RJTs are provided in Section 3.2. The *rooted treewidth* of a graph G is the minimum width of an RJT on G.

Theorem 2, which is a natural generalization of the well-known Proposition 1, ensures that, given a vector of moments on an RJT, if all these moments furthermore satisfy local independences, we can construct a distribution on the initial directed graphical model which admits these moments as marginals.

Theorem 2. *Let μ be a vector of moments in the marginal polytope of an RJT $\mathcal{G} = (\mathcal{V},\mathcal{A})$ on $G = (V,E)$. Let C be a clique, and let Z_C be the random variable on $\mathcal{X}_C$ such that $\mathbb{P}(Z_C = z_C) = \mu_C(z_C)$. Assume that, for all C, μ_C is such that $Z_v \perp\!\!\!\perp Z_{C\backslash\mathrm{des}(v)}|Z_{\mathrm{pa}(v)}$. Then the unique distribution on X_V with moments $\mu = (\mu_C)_{C\in\mathcal{V}}$ factorizes according to G.*

This result can be formulated more simply with the following simple assumption on the RJT.

Definition 2. We say that an RJT is a gradual RJT if $\forall v \in V, \mathrm{offspring}(C_v) = \{v\}$.

Informally, in a gradual RJT, each cluster contains only a single node that is not present in the parent cluster. Note that, by adding nodes to an RJT, we can always turn it into a gradual RJT. Indeed, suppose that $\mathrm{offspring}(C) = \{v_1,\dots,v_k\}$, where $v_1,\dots,v_k$ are listed in topological order. It suffices to replace the node C by $C_1 \to C_2 \to \cdots \to C_k$, where $C_i = C\backslash\{v_{i+1},\dots,v_k\}$, with an arc from the parent of C to C_1 and arcs from C_k to the children of C. Note that for a gradual RJT we have $\check{C}_v = C_v\backslash\{v\}$. In the rest of the paper, we will therefore restrict our focus to the case of gradual RJTs.

Figure 3. Example Where Satisfying (7) on Junction Tree (b) Is Not Sufficient to Ensure Factorization on Graph (a)

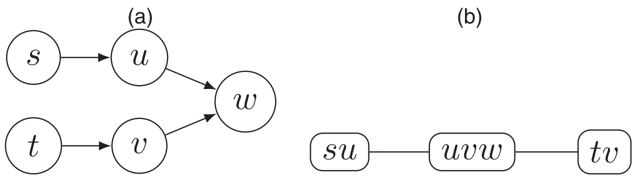
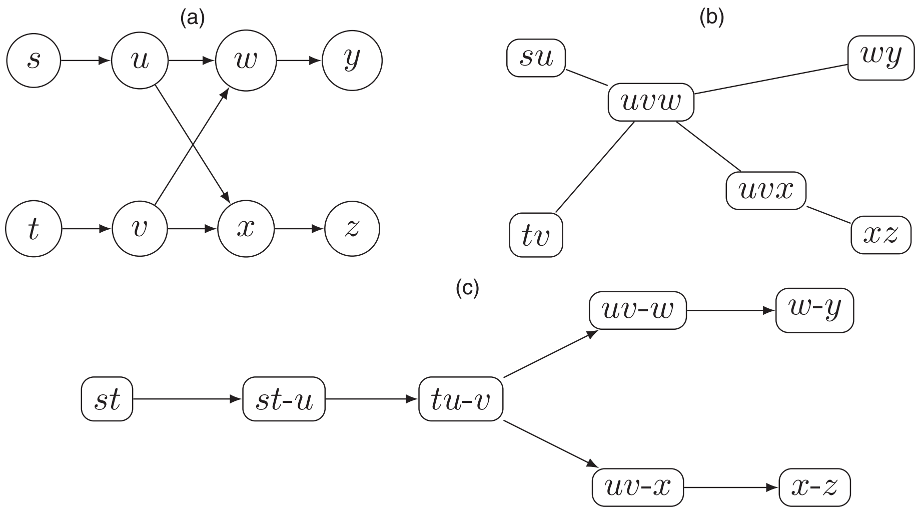


Figure 4. (a) A Directed Graph G; (b) A Junction Tree on G; (c) An RJT on G



Note. For each cluster C, we indicate on the left part of the labels the vertices of $C\backslash\text{offspring}(C)$ and on the right part the vertices of $\text{offspring}(C)$.

Corollary 3. *Let μ be a vector of moments in the marginal polytope of a* gradual *RJT $\mathcal{G} = (\mathcal{V}, \mathcal{A})$ on $G = (V, E)$. Then the unique distribution on X_V with moments $\mu = (\mu_C)_{C\in\mathcal{V}}$ that factorizes according to $\mathcal{G}$ does also factorize according to G if and only if, for all $v \in V$, all $x_{\text{pa}(v)}$ such that $\mu_{\text{pa}(v)}(x_{\text{pa}(v)}) \neq 0$, and all $x_{C_v\backslash\text{pa}(v)}$, we have*

$$\mu_{C_v}(x_{C_v}) = \mu_{v|\text{pa}(v)}(x_v|x_{\text{pa}(v)})\,\mu_{\check{C}_v}\left(x_{\check{C}_v}\right), \quad \textit{where} \quad \mu_{v|\text{pa}(v)}(x_v|x_{\text{pa}(v)}) := \frac{\mu_{\text{fa}(v)}(x_{\text{fa}(v)})}{\mu_{\text{pa}(v)}(x_{\text{pa}(v)})}.$$

Jensen et al. (1994, beginning of section 4) introduced the concept of *strong junction tree* which is similar to our concept of RJT, but they do not have the right properties for our approach.[6]

3.2. Building an RJT

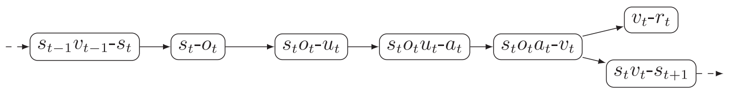
Although $(\{V\}, \emptyset)$ is an RJT, the concept has only practical interest if it is possible to construct RJTs with small cluster nodes. In that respect, note that any RJT must satisfy, for all $u, v \in V$, the implication

$$\left.\begin{array}{ll}\exists w \in V \text{ s.t. } & C_v \dashrightarrow C_w \text{ and } u \in \text{fa}(w)\\ \text{and} & C_u \dashrightarrow C_v\end{array}\right\} \Rightarrow u \in C_v, \tag{8}$$

where $C \dashrightarrow C'$ denotes the existence of a C-C' path in the RJT $\mathcal{G}$ considered. This notation will be used throughout this section. Indeed, since $u \in C_u$ and $\text{fa}(w) \subset C_w$ by definition and since $C_u \dashrightarrow C_v \dashrightarrow C_w$, the running intersection property implies $u \in C_v$. This motivates Algorithm 1, a simple gradual RJT construction algorithm which propagates iteratively elements present in each cluster node to their parent cluster node, and which thereby produces an RJT which is *minimal* in the sense that the implication in (8) is strengthened to an equivalence. It turns out that the RJT produced by Algorithm 1 has been considered in the literature under the name *bucket tree* (Kask et al. 2005, definition 5.2).[7]

The algorithm proceeds as follows. Let $\preceq$ be an arbitrary topological order on G, and let $\max_{\preceq} C$ denote the maximum of C for the topological order $\preceq$. The algorithm maintains a set C_v' for each vertex v, which coincides at the end of the algorithm with the nodes C_v in the RJT produced. We recall that $\check{C}_v'$ is the set $C_v'\backslash\{v\}$. As an illustration, for any topological order on the graph of the chess example of Figure 2(b), Algorithm 1 produces

Figure 5. RJT Produced by Algorithm 1 on the Example of Figure 2(b)



Note. The offspring of a node is to the right of symbol -.

the RJT represented in Figure 5. The following proposition shows that Algorithm 1 produces an RJT $\mathcal{G} = (\mathcal{V}, \mathcal{A})$ which is minimal for $\preceq$ in the sense that it satisfies a converse of (8).

Algorithm 1 (Create a Minimal Gradual RJT Given a Topological Order)

```
1: **Input** $G = (V, E)$ and a topological order $\preceq$ on G
2: **Initialize** $C'_v = \emptyset$ for all $v \in V$ and $\mathcal{A}' = \emptyset$
3: **for** each node $v$ of $V$ taken in reverse topological order $\preceq$ **do**
4:     $C'_v \leftarrow \mathrm{fa}(v) \cup \bigcup_{w:(v,w)\in\mathcal{A}'} \check{C}_w$
5:     **if** $\check{C}'_v \neq \emptyset$ **then**
6:         $u \leftarrow \max_{\preceq}(\check{C}_v)$          ▷ $u$ is the maximal element of $\check{C}_v \subset V$ according to $\preceq$
7:         $\mathcal{A}' \leftarrow \mathcal{A}' \cup (u, v)$
8:     **end if**
9: **end for**
10: $\mathcal{A} \leftarrow \{(C'_u, C'_v) \mid (u, v) \in \mathcal{A}'\}$
11: **Return** $\mathcal{G} = ((C'_v)_{v\in V}, \mathcal{A})$.
```

Proposition 4. *Algorithm 1 produces an RJT such that the root node C_v of v is C'_v, satisfying* $\mathrm{offspring}(C_v) = \{v\}$, *which admits $\preceq$ as a topological order, and such that $(u \in C_v) \Rightarrow (u \preceq v)$. Moreover, its cluster nodes are minimal in the sense that*

$$u \in C_v \Rightarrow \begin{cases} \exists w \in V \text{ s.t. } C_v \dashrightarrow C_w \text{ and } u \in \mathrm{fa}(w) \\ \qquad\qquad\quad C_u \dashrightarrow C_v. \end{cases} \tag{9}$$

Remark 3. Algorithm 1 takes in input a topological order on G. For practical use, we recommend using Algorithm 3 in Appendix B, which simultaneously builds the RJT and a "good" topological order.

4. MILP Formulation for IDs

Given that Algorithm 1 produces a gradual RJT we will implicitly only consider gradual RJTs from now on. In the rest of the paper, we work with the following variant of the marginal polytope $\mathcal{M}^0_{\mathcal{G}}$ defined in Equation (6):

$$\mathcal{M}_{\mathcal{G}} = \left\{ \left(\mu_{C_v}, \mu_{\check{C}_v}\right)_{v\in V} : \left(\mu_{C_v}\right)_{v\in V} \in \mathcal{M}^0_{\mathcal{G}} \text{ and } \mu_{\check{C}_v} = \sum_{x_v} \mu_{C_v} \right\},$$

where moments $\mu_{\check{C}_v}$ have been introduced. This is for convenience, and all the results could have been written using $\mathcal{M}^0_{\mathcal{G}}$.

On graphical models, the inference problem, which is hard in general, becomes easy on junction trees. Since problem (3) is NP hard even when restricted to graphs of treewidth two (Mauá et al. 2012b), unless P = NP, the situation is strictly worse for the MEU problems associated with IDs. In particular, (3) is hard for IDs whose graph G contains a directed tree with many roots,[8] but such IDs are challenging for any method because they have large cycles in their relevance graph,[9] which requires updating many policies simultaneously in algorithms like multiple policy update. However, we will see in this section that, given an RJT, we can obtain mathematical programs for MEU Problem (3) whose variables correspond to moments of the cliques of the RJT.

We first obtain an NLP formulation in Section 4.1 and then linearize it into an exact MILP in Section 4.2.

4.1. An Exact NLP Formulation

Consider a *parameterized influence diagram* (PID) encoded as the quadruple $(G, \mathcal{X}, \mathfrak{p}, r)$, where $G = (V, E)$ is a graph with set of vertices V partitioned into $(V^{\mathrm{a}}, V^{\mathrm{s}})$, with $\mathcal{X} = \prod_{v\in V} \mathcal{X}_v$ the support of the vector of random variables attached to all vertices of G, $\mathfrak{p} = \{p_{v|\mathrm{pa}(v)}\}_{v\in V^{\mathrm{s}}}$ is the collection of fixed and assumed known conditional probabilities, and $r = \{r_v\}_{v\in V^{\mathrm{r}}}$ is the collection of reward functions[10] $r_v : \mathcal{X}_v \to \mathbb{R}$ which we will also view as vectors $r_v \in \mathbb{R}^{|\mathcal{X}_v|}$.

The result of Corollary 3 together with the need to enforce the values of the conditionals $\mathfrak{p}$ motivates the following construction. For $(G, \mathcal{X}, \mathfrak{p}, r)$ a given PID and $\mathcal{G}$ a given RJT, we introduce the polytope

$$\overline{\mathcal{P}}(G, \mathcal{X}, \mathfrak{p}, \mathcal{G}) = \left\{ \mu \in \mathcal{M}_{\mathcal{G}} : \mu_{C_v} = \mu_{\check{C}_v} p_{v|\mathrm{pa}(v)} \text{ for all } v \in V^{\mathrm{s}} \right\}, \tag{10}$$

where the equality $\mu_{C_v} = \mu_{\check{C}_v} p_{v|\mathrm{pa}(v)}$ should be understood functionally, that is, meaning that $\mu_{C_v}(x_{C_v}) = \mu_{\check{C}_v}(x_{\check{C}_v}) p_{v|\mathrm{pa}(v)}(x_v|x_{\mathrm{pa}(v)}), \forall x_{C_v} \in \mathcal{X}_{C_v}$; we will use such functional (in)equalities throughout the paper. We omit the dependence of $\overline{\mathcal{P}}$ in $(G, \mathcal{X}, \mathfrak{p}, \mathcal{G})$ when the context is clear. Consider the following NLP

$$\max_{\mu,\delta} \sum_{v\in V^{\mathrm{r}}} \langle r_v, \mu_v \rangle \tag{11a}$$

$$\text{s.t.}\quad \mu \in \overline{\mathcal{P}}(G, \mathcal{X}, \mathfrak{p}, \mathcal{G}), \tag{11b}$$

$$\delta \in \Delta, \tag{11c}$$

$$\mu_{C_v} = \delta_{v|\mathrm{pa}(v)} \mu_{\check{C}_v}, \forall v \in V^{\mathrm{a}}, \tag{11d}$$

where the inner product notation $\langle r_v, \mu_v \rangle$ stands for $\sum_{x_v} \mu_v(x_v) r_v(x_v)$. Note that the constraints $\delta \in \Delta$, that is positivity and summing to 1, are implied by Constraints (11b) and (11d).[11]

By introducing the set of moments

$$\mathcal{S}_{\mathcal{G}}(G) = \left\{ \mu \in \overline{\mathcal{P}} : \exists \delta \in \Delta, \mu_{C_v} = \mu_{\check{C}_v} \delta_{v|\mathrm{pa}_G(v)} \text{ for all } v \text{ in } V^{\mathrm{a}} \right\}, \tag{12}$$

we can reformulate Problem (11) more concisely as

$$\max_{\mu \in \mathcal{S}_{\mathcal{G}}(G)} \sum_{v\in V^{\mathrm{r}}} \langle r_v, \mu_v \rangle. \tag{13}$$

As defined above, $\mathcal{S}_{\mathcal{G}}(G)$ is the set of moments corresponding to distributions induced by feasible policies: μ is in $\mathcal{S}_{\mathcal{G}}(G)$ if there exists δ in Δ such that $\mu_{C_v}(x_{C_v}) = \mathbb{P}_\delta(X_{C_v} = x_{C_v})$ for all v and x_{C_v}. We write $\mathcal{S}(G)$ when $\mathcal{G}$ is clear from context. It is nonconvex in general as shown by the examples in the proof of Theorem 9. However, we show in Section 6 that $\mathcal{S}(G)$ is a polytope if G is soluble, a property identifying "easy" IDs.

Since, in the above NLP, by Corollary 3, the equality constraints (10) and (11d) together guarantee that the associated distribution factorizes according to G and since (10) also enforces that the conditionals in $\mathfrak{p}$ are correct, we have the following result:

Theorem 5. *NLP Problems* (11) *and* (13) *are equivalent to the MEU Problem* (3) *in the sense that they have the same value and that if* (μ, δ) *is a feasible solution for Problem* (11), *then* δ *defines an admissible strategy for Problem* (3), *and* μ *characterizes the moments of the distribution induced by* δ.

Proof. If (μ, δ) is a solution of (11), then μ is a solution of (13), and conversely, if μ is a solution of (13), by the definition of $\mathcal{S}(G)$, there exists δ such that (μ, δ) is a solution of (11), which gives the equivalence between (11) and (13).

Now let (μ, δ) be an admissible solution of Problem (11). Then δ is an admissible solution of the MEU problem. We now prove that μ corresponds to the moments of the distribution $\mathbb{P}_\delta$ induced by δ, from which we can deduce that $\mathbb{E}_\delta(\sum_{v\in V^{\mathrm{r}}} r_v(X_v)) = \sum_{v\in V^{\mathrm{r}}} \langle r_v, \mu_v \rangle$. Note that if A, P, and D are disjoint subsets of V, μ is a distribution on $\mathcal{X}_V$, $\mu_{A\cup P\cup D}$ is the distribution induced by μ on $\mathcal{X}_{A\cup P\cup D}$, and $p_{D|P}$ is a conditional distribution of D given P, then

$$\mu_{A\cup P\cup D} = \mu_{A\cup P}\, p_{D|P} \qquad \Longrightarrow \qquad X_D \perp\!\!\!\perp X_A \mid X_P, \tag{14}$$

where the independence is according to μ. By (14), we have that the vector μ satisfies the conditions of Theorem 2 and, hence, corresponds to a distribution $\mathbb{P}_\mu$ that factorizes on G. Furthermore, constraint (10) ensures that $\mathbb{P}_\mu(X_v|X_{\mathrm{pa}(v)}) = p_{v|\mathrm{pa}(v)}$ for all $v \in V^{\mathrm{s}}$, which yields the result. Conversely, let δ be an admissible solution of MEU Problem (3), and let μ be the vector of moments induced by $\mathbb{P}_\delta$. We have $\mu_{C_v} = \mu_{\check{C}_v} p_{v|\mathrm{pa}(v)}$ for v in V^{s} and $\mu_{C_v} = \mu_{\check{C}_v} \delta_{v|\mathrm{pa}(v)}$ for v in V^{a}, and (μ, δ) is a solution of (11). Furthermore, $\mathbb{E}_\delta(\sum_{v\in V^{\mathrm{r}}} r_v(X_v)) = \sum_{v\in V^{\mathrm{r}}} \langle r_v, \mu_v \rangle$, and (11) is equivalent to MEU Problem (3).

4.2. MILP Formulation

NLP (11) is hard to solve due to nonlinear constraints (11d). But by Theorem 5, Problems (3) and (11) are equivalent and, in particular, admit the same optimal solutions in terms of δ.

We recall that there always exists at least one optimal strategy which is deterministic (and therefore integral) for Problem (3), that is, a strategy δ such that

$$\delta_{v|\mathrm{pa}(v)}\big(x_{\mathrm{fa}(v)}\big) \in \{0,1\}, \quad \forall x_{\mathrm{fa}(v)} \in \mathcal{X}_{\mathrm{fa}(v)}, \forall v \in V^{\mathrm{a}}. \tag{15}$$

We can therefore add integrality constraint (15) to (11). With this integrality constraint, Equation (11d) becomes a *logical constraint*, that is, a constraint of the form $\lambda y = z$ with λ binary and continuous y and z. Such constraints can be handled by modern MILP solvers, such as CPLEX or Gurobi, that can therefore directly solve Problem (11). Alternatively, by a classical result in integer programming, we can turn Problem (11) into an equivalent MILP by replacing constraint (11d) by its McCormick relaxation (McCormick 1976). For a given $\mathfrak{p}$, let b be a vector of upper bounds $b_{\check{C}_v}(x_{\check{C}_v})$ satisfying

$$\mathbb{P}_{\delta'}\left(X_{\check{C}_v} = x_{\check{C}_v}\right) \le b_{\check{C}_v}\left(x_{\check{C}_v}\right) \qquad \forall \delta' \in \Delta, \quad \forall v \in V^{\mathrm{a}}, \quad \forall x_{\check{C}_v} \in \mathcal{X}_{\check{C}_v}. \tag{16}$$

For such a vector b, we say that, for a given node v, $(\mu_{C_v}, \delta_{v|\mathrm{pa}(v)})$ satisfies McCormick's inequalities (see Appendix E) if

$$\begin{cases} \mu_{C_v} \ge \mu_{\check{C}_v} + \left(\delta_{v|\mathrm{pa}(v)} - 1\right) b_{\check{C}_v}, \\ \mu_{C_v} \le \delta_{v|\mathrm{pa}(v)}\, b_{\check{C}_v}, \\ \mu_{C_v} \le \mu_{\check{C}_v}. \end{cases} \tag{McCormick(v,b)}$$

Note that the last inequality $\mu_{C_v} \le \mu_{\check{C}_v}$ can be omitted in our case as it is implied by the marginalization constraint $\mu_{\check{C}_v} = \sum_{x_v} \mu_{C_v}$ in the definition of $\mathcal{M}_{\mathcal{G}}$. Given the upper bounds provided by b, we introduce the polytope of valid moments and decisions satisfying all McCormick constraints:

$$\mathfrak{Q}^b(G, \mathcal{X}, \mathfrak{p}, \mathcal{G}) = \left\{(\mu, \delta) \in \mathcal{M}_{\mathcal{G}} \times \Delta : \text{McCormick}(v, b) \text{ is satisfied for all } v \in V^{\mathrm{a}}\right\}. \tag{17}$$

Thus, by using McCormick on constraint (11d), we get that the MEU Problem (3) is equivalent to the following MILP:

$$\max_{\mu,\delta} \quad \sum_{v \in V^{\mathrm{r}}} \langle r_v, \mu_v \rangle \tag{18a}$$
$$\text{s.t.} \quad \mu \in \overline{\mathcal{P}}(G, \mathcal{X}, \mathfrak{p}, \mathcal{G}), \tag{18b}$$
$$\delta \in \Delta^{\mathrm{d}}, \tag{18c}$$
$$(\mu, \delta) \in \mathfrak{Q}^b, \tag{18d}$$

where Δ^{d} is the set of deterministic policies and contains the integrality constraints (15).

Remark 4. The strength of the McCormick constraints (McCormick(v,b)) depends on the quality of the bounds $b_{\check{C}_v}$ on $\mu_{\check{C}v}$. As for a solution μ of Problem (18), $\mu_{\check{C}_v}$ corresponds to a probability distribution; the simplest admissible bound over $\mu_{\check{C}_v}$ is just $b = 1$, leading to the polytope $\mathfrak{Q}^1$. Unfortunately, McCormick's constraints are loose in this case: we show in Appendix E.2.1 that, for any μ in $\overline{\mathcal{P}}$, there exists δ in Δ such that (μ, δ) satisfies those McCormick constraints. Hence, when $b = 1$, McCormick constraints fail to retain any information about the conditional independence statements encoded in the associated nonlinear constraints. Since δ does not appear outside of the McCormick constraints, their sole interest in that case is to enable the branching decisions on δ to have an impact on μ. Appendix E.2.2 gives an example showing that McCormick constraints do retain information about the conditional independence if bounds $b_{\check{C}_v}$ smaller than 1 are used. Finally, Appendix E.2.3 provides a dynamic programming algorithm that efficiently computes such a b.

5. Valid Cuts

Classical techniques in integer programming, such as branch-and-bound algorithms, rely on solving the relaxation of the MILP to obtain a lower bound on the value of the objective. For Problem (18) the relaxation is likely to be poor, and so the MILP is not well solved by off-the-shelf solvers. Indeed, as explained in Remark 4, when $b = 1$, the McCormick inequalities completely fail to capture, in the linear relaxation of MILP (18), the conditional independence encoded by nonlinear constraint (11d). Further, improving the bound b does not completely address the issue. In this section, we introduce valid cuts to strengthen the relaxation and ease the MILP resolution. Recall that a *valid cut* for a MILP is an (in)equality that is satisfied by any solution of the MILP, but not necessarily by solutions of its linear relaxation. A family of valid cuts is stronger than another when the former yields a polytope strictly included in the latter.

5.1. Constructing Valid Cuts

By restricting ourselves to vectors of moments $\mu \in \overline{\mathcal{P}}$, we have imposed

$$\mathbb{P}(X_v|X_{V\backslash \mathrm{des}(v)}) = \mathbb{P}\Big(X_v|X_{\check{C}_v}\Big) = p_{v|\mathrm{pa}(v)} \quad \text{for all } v \text{ in } V^{\mathrm{s}},$$

because $\mu \in \overline{\mathcal{P}}$ must satisfy $\mu_{C_v} = \mu_{\check{C}_v} p_{v|\mathrm{pa}(v)}$. If we could impose as well the nonlinear constraints $\mu_{C_v} = \mu_{\check{C}_v} \delta_{v|\mathrm{pa}(v)}$ for v in V^{a}, we would be able to impose that decisions encoded in μ at the nodes $a \in V^{\mathrm{a}}$ satisfy $\mathbb{P}(X_a|X_{C_a\backslash\{a\}}) = \mathbb{P}(X_a|X_{\mathrm{pa}(a)})$, and thanks to Corollary 3, this would be sufficient to guarantee that the distribution with moments $(\mu_{C_v})_{v\in V}$ factorizes according to G. A key question is therefore whether we can enforce some conditional independence implied by the nonlinear constraints, and thus lost in the linear relaxation, through linear constraints. This seems possible, because as an indirect consequence of setting the conditional distributions $p_{v|\mathrm{pa}(v)}$ for $v \in V^{\mathrm{s}}$, there are other conditional distributions whose value does not depend on δ. Indeed, thanks to the structure of the graph G, for some pairs of sets of vertices C, D with $D \subseteq C$, the conditional probabilities $\mathbb{P}_\delta(X_D = x_D|X_{C\backslash D} = x_{C\backslash D})$ are identical[12] for any strategy δ and can be precomputed using a classical probabilistic inference algorithm on the graph. We can thus add linear constraints, also known as valid cuts, of the form

$$\mu_C = \mu_{C\backslash D}\, p_{D|C\backslash D}. \tag{19}$$

Although these additional constraints are not needed to set the value of the conditionals on $v \in V^{\mathrm{s}}$ and the conditional independences of the form $X_v \perp\!\!\!\perp X_{V\backslash\mathrm{des}(v)} \mid X_{\mathrm{pa}(v)}$ for $v \in V^{\mathrm{s}}$, they can be useful to enforce some of the conditional independences that should be satisfied by μ at decision nodes. In particular, if there exists a strict subset M of $C\backslash D$ such that $p_{D|C\backslash D} = p_{D|M}$, then (19) enforces that, for any $v \in V^{\mathrm{a}} \cap (C\backslash(D\cup M))$, we have $\mathbb{P}(X_a|X_{D\cup M}) = \mathbb{P}(X_a|X_M)$. Clearly, the larger the D, the stronger the valid cut. This motivates the following definition.

Definition 3. *Given a set of vertices C, we define $C^{\perp\!\!\!\perp}$ to be the largest subset D of C such that, for any parameterization of G, there exists $p_{D|C\backslash D}$ such that $\mathbb{P}_\delta(X_D|X_{C\backslash D}) = p_{D|C\backslash D}$ regardless of the strategy δ. We define $C^{\not\perp\!\!\!\perp}$ as $C\backslash C^{\perp\!\!\!\perp}$.*

It is not obvious that a largest such set exists and is unique and, therefore, that $C^{\perp\!\!\!\perp}$ is well defined. We fully characterize $C^{\perp\!\!\!\perp}$ and prove its existence in Appendix C.1. Given that $C^{\perp\!\!\!\perp}$ is well defined, the equalities

$$\mu_C = \mu_{C^{\not\perp\!\!\!\perp}} p_{C^{\perp\!\!\!\perp}|C^{\not\perp\!\!\!\perp}}, \quad \forall C \in \mathcal{V}, \tag{20}$$

are the strongest valid cuts of the form (19) that we can obtain for Problem (18). For each C, Cohen and Parmentier (2019) show that $C^{\perp\!\!\!\perp}$ can even be computed in $O(|C||E|)$ time, and we believe it can be computed in $O(|E|)$ time, so to compute $(C_v^{\perp\!\!\!\perp})_{v\in V^{\mathrm{a}}}$, the total complexity is between $O(|V^{\mathrm{a}}||V|)$ and $O(\max_{v\in V^{\mathrm{a}}}|C_v||V^{\mathrm{a}}||V|)$. Then $(\mu_C)_{C\in\mathcal{V}}$ can be computed in $O(\kappa^\omega|\mathcal{V}|)$ time (which is $O(\kappa^\omega|V|)$ for a graded RJT), with $\omega = \max_{C\in\mathcal{V}}|C|$ the *width* of the RJT and $\kappa = \max_{v\in V}|\mathcal{X}_v|$. In our experiments, the time to compute all the $C^{\perp\!\!\!\perp}$ and all the $p_{C^{\perp\!\!\!\perp}|C^{\not\perp\!\!\!\perp}}$ was negligible compared with the time needed to solve a linear program or the MILP.

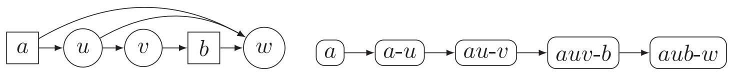
We can then define $\mathcal{P}^{\perp\!\!\!\perp}$ as the polytope we obtain when we strengthen $\overline{\mathcal{P}}$ with our valid cuts:

$$\mathcal{P}^{\perp\!\!\!\perp}(G,\mathcal{X},\mathfrak{p},\mathcal{G}) = \left\{\mu \in \overline{\mathcal{P}} : \mu_{C_v} = p_{C_v^{\perp\!\!\!\perp}|C_v^{\not\perp\!\!\!\perp}} \sum_{x_{C_v^{\perp\!\!\!\perp}}} \mu_{C_v} \text{ for all } v \in V^{\mathrm{a}}\right\}. \tag{21}$$

In the definition of $\mathcal{P}^{\perp\!\!\!\perp}$, we decided to introduce valid cuts of the form (20) only for sets of vertices C of the form C_v with $v \in V^{\mathrm{a}}$. This is to strike a balance between the number of constraints added and the number of independences enforced. Our choice is however heuristic, and it could notably be relevant to introduce constraints of the form (20) for well-chosen $C \subsetneq C_v$.

Figure 6 provides an example of an ID where the introduction of a valid cut of the form (20) reduces the size of the initial polytope. Indeed, the cluster $C = \{a,u,v,b\}$ leads to $C^{\perp\!\!\!\perp} = \{u\}$, and the resulting cut (20) is not implied by the linear inequalities of (18). Indeed, suppose that $\mathcal{X}_a = \mathcal{X}_v = \{0\}$, whereas $\mathcal{X}_u = \mathcal{X}_b = \{0,1\}$.

Figure 6. ID and Its RJT with a Nonvalid Cut (20) for $C = \{a,u,v,b\}$ with $C^{\perp\!\!\!\perp} = \{u\}$



Then the solution defined by $\mu_{auvb}(0,i,0,i) = 0.5$ and $\mu_{auvb}(0,i,0,1-i) = 0$ for $i \in \{0,1\}$ is in the linear relaxation of (18) but does not satisfy (20). To compute $C^{\perp\!\!\!\perp}$, we have used the characterization provided in Appendix C.1.

5.2. Strength of the Relaxations and Their Interpretation in Terms of Graph

Consider (G,ρ), a PID with $G = (V^{\mathrm{s}}, V^{\mathrm{a}}, E)$ and $\rho = (\mathscr{X}, \mathfrak{p}, r)$. Let $\mathscr{G}$ be an RJT on G, and let b be an admissible bound satisfying (16). The valid cuts of Section 5.1 enable the introduction of the following strengthened version of MILP (18):

$$\max_{\mu,\delta} \sum_{v\in V^{\mathrm{r}}} \langle r_v, \mu_v\rangle \text{ subject to } \mu \in \mathscr{P}^{\perp\!\!\!\perp}(G,\mathscr{X},\mathfrak{p},\mathscr{G}), \quad \delta \in \Delta^{\mathrm{d}}, \quad (\mu,\delta) \in \mathfrak{Q}^b. \tag{22}$$

The following proposition summarizes the results of Section 5.1.

Proposition 6. *Any feasible solution (μ,δ) of MILP (22) is such that μ is the vector of moments of the distribution $\mathbb{P}_\delta$. Hence, (μ,δ) is an optimal solution of (22) if and only if δ is an optimal deterministic solution of the MEU problem (3) on (G,ρ).*

In this section, we provide interpretations of the linear relaxations of (18) and (22) in terms of graphs. We introduce the sets of arcs and IDs

$$\begin{aligned} \overline{E} &= E \cup \{(u,v) : v \in V^{\mathrm{a}} \text{ and } u \in C_v \backslash \mathrm{fa}(v)\}, & \overline{G} &= (V^{\mathrm{s}}, V^{\mathrm{a}}, \overline{E}), \\ E^{\perp\!\!\!\perp} &= E \cup \{(u,v) : v \in V^{\mathrm{a}} \text{ and } u \in C_v^{\not\perp\!\!\!\perp} \backslash \mathrm{fa}(v)\}, \quad \text{and} & G^{\perp\!\!\!\perp} &= (V^{\mathrm{s}}, V^{\mathrm{a}}, E^{\perp\!\!\!\perp}). \end{aligned}$$

Figure 7 illustrates $\overline{G}$ and $G^{\perp\!\!\!\perp}$ on the ID of Figure 2(b). Note that $E \subseteq E^{\perp\!\!\!\perp} \subseteq \overline{E}$, and note the following three facts on $\overline{G}$ and $G^{\perp\!\!\!\perp}$. First, the definition of both IDs depends on G and $\mathscr{G}$. Second, $\mathscr{G}$ is still an RJT on $\overline{G}$ and $G^{\perp\!\!\!\perp}$. Third, any parameterization $(\mathscr{X}_V, \mathfrak{p}, r)$ of G is also a parameterization of $\overline{G}$ and of $G^{\perp\!\!\!\perp}$. The second and third results are satisfied by any ID $G' = (V^{\mathrm{s}}, V^{\mathrm{a}}, E \cup E')$, where E' contains only arcs of the form (u,v) with $v \in V^{\mathrm{a}}$ and $u \in C_v$. Hence, if we denote by $\Delta_{G'}$ the set of feasible strategies for $(G', \mathscr{X}_V, \mathfrak{p}, r)$, we can extend the definition of $\mathscr{S}(G)$ in Equation (12) to such G' with

$$\mathscr{S}(G') = \left\{\mu \in \overline{\mathscr{P}} : \exists \delta \in \Delta_{G'}, \mu_{C_v} = \mu_{\check{C}_v} \delta_{v|\mathrm{pa}_{G'}(v)} \text{ for all } v \text{ in } V^{\mathrm{a}}\right\}.$$

Theorem 7. *We have*

$$\overline{\mathscr{P}} = \mathscr{S}(\overline{G}) \quad \text{and} \quad \max_{\mu\in\overline{\mathscr{P}}} \sum_{v\in V^{\mathrm{r}}} \langle r_v, \mu_v\rangle = MEU(\overline{G},\rho),$$

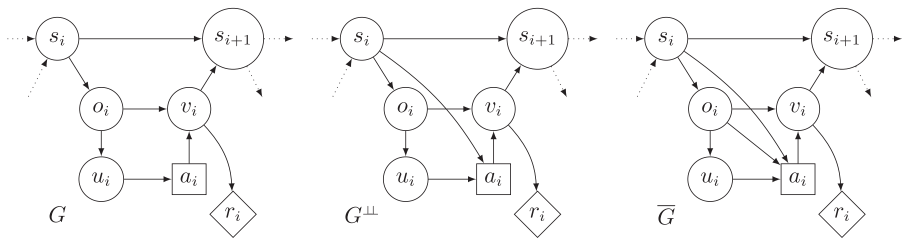
and

$$\mathscr{P}^{\perp\!\!\!\perp} = \mathscr{S}(G^{\perp\!\!\!\perp}) \quad \text{and} \quad \max_{\mu\in\mathscr{P}^{\perp\!\!\!\perp}} \sum_{v\in V^{\mathrm{r}}} \langle r_v, \mu_v\rangle = MEU(G^{\perp\!\!\!\perp},\rho).$$

Hence, if (μ,δ) is a solution of the linear relaxation of (18), then δ is a strategy on $\overline{G}$, whereas if (μ,δ) is a solution of the linear relaxation of (22), then δ is a strategy on $G^{\perp\!\!\!\perp}$.

Furthermore, note that $\mathscr{S}(G')$ is generally not a polytope. Indeed, when $G' = G$, this is the reason why (11) is not a linear program. An important result of the theorem is that $\mathscr{S}(\overline{G})$ and $\mathscr{S}(G^{\perp\!\!\!\perp})$ are polytopes, and $MEU(\overline{G},\rho)$ and $MEU(G^{\perp\!\!\!\perp},\rho)$ can therefore be solved using the linear programs $\max_{\mu\in\overline{\mathscr{P}}} \sum_{v\in V^{\mathrm{r}}} \langle r_v, \mu_v\rangle$ and $\max_{\mu\in\mathscr{P}^{\perp\!\!\!\perp}} \sum_{v\in V^{\mathrm{r}}} \langle r_v, \mu_v\rangle$, respectively. The proof of the theorem can be found in Appendix C.2.

Figure 7. Soluble Relaxations Corresponding to Linear Relaxations for the Chess Example



6. Soluble IDs

In this section, we make the assumption that IDs are such that any vertex $v \in V$ has a descendant in the set of utility vertices V^{r}, that is, $V^{\mathrm{s}} \cup V^{\mathrm{a}} = \overline{\mathrm{anc}}(V^{\mathrm{r}})$. The following remark explains why we can make this assumption without loss of generality.

Remark 5. Consider a parameterized ID (G, ρ), where $G = (V^{\mathrm{s}}, V^{\mathrm{a}}, E)$ and V^{s} is the union of chance vertices V^{c} and utility vertices V^{r}. Let (G', ρ') be the ID obtained by removing any vertex that is not in V^{r} and has no descendant in V^{r}, and restrict ρ accordingly. If a random vector X_V factorizes as a directed graphical model on (V, E) and $V' \subseteq V$ is such that $\overline{\mathrm{anc}}(V') = V'$, then $X_{V'}$ factorizes as a directed graphical model on the subgraph induced by V' with the same conditional probabilities $p_{v|\mathrm{pa}(v)}$. Hence, given a strategy δ on (G, ρ) and its restriction δ' to (G', ρ'), we have $\mathbb{E}_\delta(\sum_{v\in V^{\mathrm{r}}} r_v(X_v)) = \mathbb{E}_{\delta'}(\sum_{v\in V^{\mathrm{r}}} r_v(X_v))$, where the first expectation is taken in (G, ρ) and the second in (G', ρ'), and the two IDs model the same MEU problem.

The proofs of this section are quite technical and can be found in Appendix D.

6.1. Linear Program for Soluble IDs

Consider an ID $G = (V^{\mathrm{s}}, V^{\mathrm{a}}, E)$ with $V^{\mathrm{s}} = V^{\mathrm{c}} \cup V^{\mathrm{r}}$. Given a strategy $(\delta_u)_{u\in V^{\mathrm{a}}}$ and a decision vertex v, we denote by δ_{-v} the partial strategy $(\delta_u)_{u\in V^{\mathrm{a}}\backslash v}$. A strategy $(\delta_v)_{v\in V^{\mathrm{a}}}$ is called a *local optimum* if

$$\delta_v \in \operatorname*{arg\,max}_{\delta'_v \in \Delta_v} \mathbb{E}_{\delta'_v, \delta_{-v}}\left(\sum_{u\in V^{\mathrm{r}}} r_u(X_u)\right) \quad \text{for each vertex } v \text{ in } V^{\mathrm{a}}.$$

It is a *global optimum* if it is an optimal solution of (3). Three concepts, *strategic relevance*, *s-reachability*, and the *relevance graph* have been introduced in the literature to characterize when a local minimum is also global (see, e.g., Koller and Friedman 2009, chapter 23.5). A decision vertex v *strategically relies* on u if the choice of a locally optimal policy δ_v given $(\delta_w)_{w\neq v}$ depends on δ_u for some parameterization ρ. As we detail at the beginning of Appendix D, *s-reachability* is a graphical characterization of strategic relevance which can be checked in linear time in the size of G: If u is not s-reachable from v, then v does not strategically rely on u, whereas if u is s-reachable from v, then there exists a parameterization ρ such that v strategically relies on u. The *relevance graph* of G is the digraph H with vertex set V^{a} and whose arcs are the pairs (v, u) of decision vertices such that u is s-reachable from v. Finally, the *SPU* algorithm (Lauritzen and Nilsson 2001) is the standard coordinate ascent heuristic for IDs. It iteratively improves a strategy δ as follows: At each step, a vertex v is picked, and δ_v is replaced by an element in $\arg\max_{\delta'_v\in\Delta_v} \mathbb{E}_{\delta'_v,\delta_{-v}}(\sum_{u\in V^{\mathrm{r}}} r_u(X_u))$. The following proposition characterizes a subset of IDs, called *soluble IDs*, which are easily solved, and provides several equivalent criteria to identify them.

Proposition 8 (Koller and Friedman 2009, theorem 23.5)**.** *Given an ID G, the following statements are equivalent and define a soluble ID.*

1. *For any parameterization ρ of G, any local optimum is a global optimum.*
2. *For any parameterization ρ of G, SPU converges to a global optimum in a finite number of steps.*[13]
3. *The relevance graph of G is acyclic.*

Given a parameterized ID G and an RJT $\mathcal{G}$, we introduced in Equation (12) the notation $\mathcal{S}(G)$ for the subset of the marginal polytope $\mathcal{M}_G$ corresponding to moments of policies. The following theorem introduces a new characterization of soluble IDs in terms of convexity.

Theorem 9. *If G is not soluble, then there exists a parameterization ρ such that, for any junction tree $\mathcal{G}$, the set of achievable moments $\mathcal{S}_{\mathcal{G}}(G)$ is not convex. If G is soluble, Algorithm 2 returns an RJT such that $\mathcal{P}^{\perp\!\!\perp} = \mathcal{S}_{\mathcal{G}}(G)$ for any parameterization ρ.*

The property of being soluble characterizes "easy" IDs that can be solved by SPU. Theorems 5 and 9 imply that if G is soluble, then our MILP formulation (22) reduces to the linear program

$$\max_{\mu\in\mathcal{P}^{\perp\!\!\perp}} \sum_{v\in V^{\mathrm{r}}} \langle r_v, \mu_v \rangle$$

and is therefore "easy" to solve. Of course, this property of being "easy" refers only to the decision part of the ID. If a soluble ID is such that, given a strategy, the inference problem is not tractable, then both SPU and our MILP formulation will not be tractable in practice. Theorem 9 is a corollary of Theorem 7 and the following lemma, and both results are proved in Appendix D.

Lemma 10. *There exists an RJT $\mathcal{G}$ such that $G^{\amalg} = G$ if and only if G is soluble. Such an RJT can be computed using Algorithm 2.*

Note that, based on a topological order of the relevance graph, Algorithm 2 proceeds by computing a larger graph (satisfying perfect recall) that contains the graph G and that assigns the same parent sets to elements of V^{s} as in G, and then uses a topological order of this graph to order the nodes of G for the computation of an RJT.

Algorithm 2 (Build a "Good" RJT for a Soluble Graph G)

```
 1: **Input:** An ID $G = (V^{\mathrm{s}}, V^{\mathrm{a}}, E)$.
 2: **Initialize:** $E' = \emptyset$.
 3: Compute the relevance graph $H$ of $G$
 4: Compute an arbitrary topological order $\preceq_H$ on $V^{\mathrm{a}}$ for the relevance graph $H$
 5: $E' \leftarrow E \cup \{(u,v) \in V^{\mathrm{a}} \times V^{\mathrm{a}} : u \preceq_H v\}$
 6: $G' \leftarrow (V, E')$
 7: $E'' \leftarrow E \cup \{(u,v) \in V^{\mathrm{s}} \times V^{\mathrm{a}} : u \notin \mathrm{des}_{G'}(v)\}$
 8: $G'' = (V, E'')$
 9: Compute an arbitrary topological order $\preceq$ on $G''$
10: **Return** the result of Algorithm 1 for $(G, \preceq)$
```

6.2. Comparison of Soluble and Linear Relaxations

MILP solvers are based on (much improved) branch-and-bound algorithms that use the linear relaxation to obtain bounds. Their ability to solve formulation (22) therefore depends on the quality of the bound provided by the linear relaxation. Since SPU solves efficiently soluble IDs, we could imagine alternative branch-and-bound schemes that use bounds computed using SPU on an ID larger than G which is soluble. To formalize this idea, we introduce the following notion: A *soluble graph relaxation* of an ID $G = (V^{\mathrm{s}}, V^{\mathrm{a}}, E)$ is a soluble ID $G' = (V^{\mathrm{s}}, V^{\mathrm{a}}, E')$, where E' is the union of E and a set of arcs with head in V^{a}.

Note that Theorem 7 can be reinterpreted as the link between soluble graph relaxation and linear relaxations. And since $\mathcal{S}(\overline{G}) = \overline{\mathcal{P}}$ and $\mathcal{S}(G^{\amalg}) = \mathcal{P}^{\amalg}$, by Theorem 9, $G^{\amalg}$ and $\overline{G}$ are soluble and, therefore, soluble graph relaxations of G.

Since any feasible strategy for the ID G is a feasible strategy for a soluble graph relaxation G', for any parameterization ρ, the value of $MEU(G', \rho)$, which can be computed by SPU, provides a tractable bound on $MEU(G, \rho)$. Soluble relaxations can therefore be used in branch-and-bound schemes for IDs, as proposed in Khaled et al. (2013). To compare the relevance of such a scheme to our MILP approach we need to compare the quality of the soluble graph relaxation and linear relaxation bounds.

Corollary 11. *Let G' be a soluble graph relaxation of G, and let $\mathcal{G}$ be the RJT obtained by running Algorithm 2 on G'. Then the linear relaxation of (22) applied to G with RJT $\mathcal{G}$ provides a bound at least as good as the one provided by the soluble relaxation G'.*

Note that this bound can sometimes be strictly better thanks to constraints $(\mu, \delta) \in \mathcal{Q}^b$.

Remark 6. In the literature, soluble relaxations have already been used to obtain bounds in different settings. For example, Yuan et al. (2010) used them in a branch-and-bound scheme. Their bounds rely on the notion of *sufficient information set* (SIS) for a decision node v (Nilsson and Höhle 2001).[14] SISs are sets of nodes that have the following property: if, given an ID G, we have a SIS D_v for each decision node v, then the ID we obtain when we add arcs u, v for each u in D_v is a soluble relaxation of G. Different SISs may be available for a given decision vertex. Poh and Horvitz (1996, theorem 2) show that the closer the SIS is to the descendant of the vertex, the worse the bound is, but the easier the inference is. Yuan et al. (2010) make the choice of an easy inference and propose to use a SIS of minimum cardinality. An alternative option would be to add the perfect recall arcs, which would lead to a much harder inference problem but to better bounds. Using our $G^{\amalg}$ corresponds to the following choice: among the SISs that enable to use our RJT for inference, use the one that leads to the best relaxation.

7. Numerical Experiments

In this paper, we have introduced MILP formulation (18) for the MEU problem and shown with Corollary 11 that, when strengthened with valid inequalities and well-chosen bounds in the McCormick constraints, the bounds provided by the linear relaxation of our formulations are better than the ones obtained by the soluble relaxations used in the literature. In this section, we study how these formulations behave numerically on instances of Examples 1 and 3.

Our formulations should not be seen as an alternative to SPU since they have different objectives. SPU is a heuristic that enables one to find quickly a good solution, which is generally a local optimum on our instances because these are not soluble IDs. Our exact approaches are an order of magnitude slower than SPU but find better solutions than SPU and prove small optimality gaps. It is therefore natural to use the two approaches sequentially and warm start the MILP solver with the solution returned by SPU, which we do in all our numerical experiments.

Given their importance to reduce the optimality gaps, we carefully study the impact of our valid inequalities on the linear relaxation bound. For notational simplicity and since it is unambiguous, in the rest of this section, we use the same notation to refer to a given node of the graph and to refer to the random variable associated with this node.

7.1. Experimental Settings

7.1.1. Experiments Performed on Each Instance. We have introduced two elements to strengthen the linear relaxation of our MILP formulation. We remind the reader that we introduced in Equation (17) the polytope $\mathcal{Q}^b$ obtained using the vector of bounds b in McCormick constraints. Also, recall from Remark 4 that, in the special case of the polytope $\mathcal{Q}^1$ obtained with $b = 1$, McCormick constraints are loose. To study the impact of McCormick constraints and valid inequalities, we solve the problems $\max\{\sum_{v\in V^{\mathrm{r}}}\langle r_v, \mu_v\rangle \mid (\mu,\delta)\in\mathcal{Q}, \delta\in\Delta^{\mathrm{d}}\}$ with four different sets $\mathcal{Q}$: $\overline{\mathcal{Q}}^1 = (\overline{\mathcal{P}}\times\Delta^d)\cap\mathcal{Q}^1$ (no cuts), $\overline{\mathcal{Q}}^b = (\overline{\mathcal{P}}\times\Delta^d)\cap\mathcal{Q}^b$ (McCormick only), $\mathcal{Q}^{\perp\!\!\!\perp,1} = (\mathcal{P}^{\perp\!\!\!\perp}\times\Delta^d)\cap\mathcal{Q}^1$ (independence cuts only), and $\mathcal{Q}^{\perp\!\!\!\perp,b} = (\mathcal{P}^{\perp\!\!\!\perp}\times\Delta^d)\cap\mathcal{Q}^b$ (McCormick and independence cuts).

7.1.2. Instances Considered. Examples 1 and 3 are multistage models. Let T denote the number of time steps of an instance. Once T has been chosen, the ID G is known, and all that is left to do is to choose a parameterization ρ. We consider instances such that, for all $v\in\mathrm{fa}(V^{\mathrm{a}})$, $\mathcal{X}_v$ has k_a elements, and for all $v\in V\backslash\mathrm{fa}(V^{\mathrm{a}})$, $\mathcal{X}_s$ has k_s elements. As we explain in the next paragraph, $\mathrm{k}_s, \mathrm{k}_a$, and T control how hard the problem is. To generate a PID instance, we start by choosing $(\mathrm{k}_s, \mathrm{k}_a, T)$, which also sets the ID. Then we draw uniformly on $[0,1]$ the conditional probabilities $p_{v|\mathrm{pa}(v)}$ for all $v\in V\backslash V^{\mathrm{a}}$ and $x_{\mathrm{fa}(v)}\in\mathcal{X}_{\mathrm{fa}(v)}$, and we renormalize. We draw uniformly on $[0,10]$ the rewards $r_v(x_v)$ for all $v\in V^{\mathrm{r}}$ and all $x_v\in\mathcal{X}_v$. For our results to be representative of any instance with parameters $(\mathrm{k}_s, \mathrm{k}_a, T)$, we generate 50 instances for each triplet and report averaged results over these 50 instances.

7.1.3. Intrinsic Difficulty of the Instances Considered. Solving an ID requires finding an optimal strategy, which is difficult because evaluating a given strategy is already difficult in the first place, and because optimizing on the set of strategies is then also difficult. The difficulty of evaluating a strategy is the difficulty of solving an inference problem on the underlying graph. A good indicator of this difficulty is therefore the treewidth of the graph. There is no measure that characterizes the intrinsic complexity of the problem of finding an optimal strategy, but the cost of the naive approach is the number of feasible deterministic strategies (Mauá et al. 2012b), that is, $|\Delta^{\mathrm{d}}|$. Our instances have a moderate treewidth, two for Example 1 and three for Example 3, and are therefore not difficult from an inference point of view. But they could be a priori difficult from an optimization point of view, because $\left|\Delta^{\mathrm{d}}\right| = \prod_{v\in V^{\mathrm{a}}}|\mathcal{X}_v|^{\prod_{u\in\mathrm{pa}(v)}|\mathcal{X}_u|} = \mathrm{k}_a^{T\mathrm{k}_s}$ is large.

7.1.4. Size of Our MILP Formulations on the Instances Considered. The number of constraints and variables in our MILP[15] is in $O(|V|\kappa^{\omega_{\mathrm{r}}+1})$, where ω_{r} is the rooted treewidth of the ID and $\kappa = \max_{v\in V}|\mathcal{X}_v|$. Our MILP formulations can therefore only deal with instances of moderate rooted treewidth, which can be arbitrarily larger than the treewidth. In our examples, the rooted treewidth is equal to the treewidth and no greater than three while $\kappa = \max(\mathrm{k}_s, \mathrm{k}_a)$, and so the size of the MILPs remains tractable for instances with large $|\Delta^{\mathrm{d}}|$.

7.1.5. Experimental Settings. All MILPs have been written in Julia (Bezanson et al. 2017) with the JuMP (Dunning et al. 2017) interface and solved using Gurobi 7.5.2 with the default settings. Experiments have been run on a server with 192 Gb of RAM and 32 cores at 3.30 GHz.

7.1.6. Reported Results. The numerical results obtained on Examples 1 and 3 are reported in Table 1. We denote by z, z^{LR}, and z^{B} the value of the best integer solution found, the optimal value of the linear relaxation, and the best upper bound found, respectively. We define the integrality gap g_{i} as $\frac{z^{\mathrm{LR}}-z}{z^{\mathrm{LR}}}$ and the final

Table 1. Mean Results on 50 Randomly Generated Instances with a Time Limit of 3,600 s

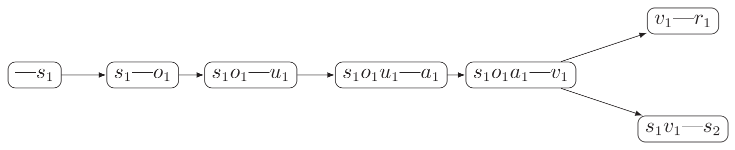
			Example 3: Chess game					Example 1: POMDP				
$(\mathrm{k}_s, \mathrm{k}_a, T)$	$\lvert\Delta^{\mathrm{d}}\rvert$	Polytope	g_{i}(%)	g_{f}(%)	i_{SPU}(%)	Opt(%)	Time(s)	g_{i}(%)	g_{f}(%)	i_{SPU}(%)	Opt(%)	Time(s)
(3,5,20)	10^{69}	$\overline{\mathcal{Q}}^1$	6.33	0.41	0.04	52	1,826	8.64	4.69	0.99	8	3,094
		$\overline{\mathcal{Q}}^b$	5.73	0.42	0.04	54	1,807	8.11	4.42	0.97	16	3,071
		$\mathcal{Q}^{\perp\!\!\!\perp,1}$	1.30	0.22	0.04	68	1,405	2.33	1.30	0.99	36	2,431
		$\mathcal{Q}^{\perp\!\!\!\perp,b}$	1.20	0.20	0.04	68	1,331	1.96	1.28	1.00	36	2,376
(3,6,20)	10^{93}	$\overline{\mathcal{Q}}^1$	5.69	0.66	0.05	27	2,702	9.66	5.88	1.99	10	3,244
		$\overline{\mathcal{Q}}^b$	5.14	0.66	0.05	31	2,612	9.08	5.56	1.99	12	3,235
		$\mathcal{Q}^{\perp\!\!\!\perp,1}$	1.02	0.32	0.05	45	2,126	2.54	1.57	2.00	20	2,931
		$\mathcal{Q}^{\perp\!\!\!\perp,b}$	0.95	0.30	0.05	47	2,112	2.08	1.52	2.00	24	2,800
(3,9,20)	10^{171}	$\overline{\mathcal{Q}}^1$	6.49	1.79	0.09	6	3,433	8.15	5.99	1.85	4	3,514
		$\overline{\mathcal{Q}}^b$	5.94	1.90	0.07	6	3,476	7.68	5.73	1.88	4	3,477
		$\mathcal{Q}^{\perp\!\!\!\perp,1}$	1.54	0.84	0.09	18	2,974	2.22	1.65	1.85	14	3,124
		$\mathcal{Q}^{\perp\!\!\!\perp,b}$	1.45	0.83	0.07	14	3,103	1.83	1.59	1.85	16	3,087
(3,10,20)	10^{200}	$\overline{\mathcal{Q}}^1$	6.85	2.36	0.06	2	3,540	10.99	8.68	1.21	4	3,469
		$\overline{\mathcal{Q}}^b$	6.31	2.08	0.05	6	3,421	10.30	8.50	1.27	4	3,467
		$\mathcal{Q}^{\perp\!\!\!\perp,1}$	1.54	0.99	0.07	14	3,110	3.32	2.74	1.20	8	3,314
		$\mathcal{Q}^{\perp\!\!\!\perp,b}$	1.45	0.96	0.06	14	3,110	2.80	2.72	1.21	8	3,314
(4,9,20)	10^{171}	$\overline{\mathcal{Q}}^1$	7.44	4.04	0.04	0	>3,600	10.27	7.94	0.98	0	>3,600
		$\overline{\mathcal{Q}}^b$	6.99	4.15	0.04	0	>3,600	9.57	7.77	1.00	0	>3,600
		$\mathcal{Q}^{\perp\!\!\!\perp,1}$	1.74	1.15	0.04	10	3,240	2.85	2.18	1.00	10	3,274
		$\mathcal{Q}^{\perp\!\!\!\perp,b}$	1.64	1.15	0.04	10	3,240	2.26	2.14	0.99	10	3,263
(4,10,20)	10^{200}	$\overline{\mathcal{Q}}^1$	8.07	4.20	0.08	0	>3,600	12.8	10.5	0.81	0	>3,600
		$\overline{\mathcal{Q}}^b$	7.62	4.65	0.04	0	>3,600	12.0	10.4	0.86	0	>3,600
		$\mathcal{Q}^{\perp\!\!\!\perp,1}$	1.31	1.31	0.07	5	3,411	4.1	3.5	0.41	0	>3,600
		$\mathcal{Q}^{\perp\!\!\!\perp,b}$	1.29	1.29	0.08	5	3,410	3.4	3.4	0.51	0	>3,600

gap g_{f} as $\frac{z^{\mathrm{B}}-z}{z^{\mathrm{B}}}$. Let z^{SPU} be the value obtained using SPU. We define the improvement with respect to SPU i_{SPU} as $i_{\mathrm{SPU}} = \frac{z-z^{\mathrm{SPU}}}{z^{\mathrm{SPU}}}$. Each line in Table 1 provides average values of different quantities on 50 instances with identical parameter $(\mathrm{k}_s, \mathrm{k}_a, T)$. The first column specifies the value of $(\mathrm{k}_s, \mathrm{k}_a, T)$ for the instances considered; the second column specifies the approximate number of admissible strategies. The third column indicates the cuts used. In the next three columns, we report the average value of g_{i}, g_{f}, and i_{SPU} on the 50 instances considered. Column "Opt" provides the percentage of instances solved to optimality, and column "Time" provides the average computing time. All gaps are given in percent, and computing times are given in seconds. Sometimes, the time limit is reached only for some of the 50 instances, and we end up with a nonzero average final gap together with an average computing time that is smaller than the time limit.

7.2. Bob and Alice Daily Chess Game

We consider the chess game example represented in Figure 2(b). The beginning of the RJT built by Algorithm 1 for this example is represented in Figure 8. The rooted treewidth of this problem is three. Table 1 reports results on the generated instances. We can tackle large instances of this problem. We can reach optimality in less than one hour for a strategy set of size 10^{171} and find a small provable gap on even bigger instances. Moreover, we see that the independence cuts enable one to strongly reduce the gaps and the computing time, whereas the improved McCormick bounds yield more minor improvements. However, on this problem, our MILP formulation only marginally improves the results returned by SPU, and its main value is the bound obtained.

Figure 8. RJT for the Chess Game



Note. The element to the right of — is the offspring.

Figure 9. RJT Built by Algorithm 1 for a POMDP with Limited Memory

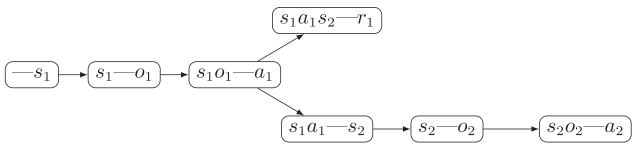
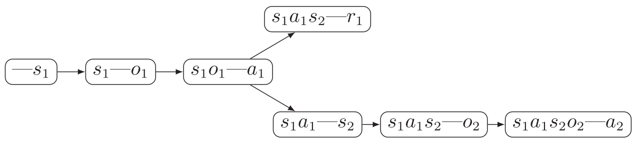


Figure 10. A Bigger RJT for a POMDP with Limited Memory



7.3. POMDP with Limited Memory

We now consider our POMDP instances. Figure 1(b) provides the graph representation of the POMDP with limited information. The rooted treewidth of this problem is two. This ID is not soluble.[16] Figure 9 represents the RJT built by Algorithm 1. On this RJT, $G^{\perp\!\!\perp} = \overline{G}$, and thus, $\mathcal{P}^{\perp\!\!\perp} = \overline{P}$, which is also the constraint set of the classical MDP relaxation of a POMDP, in which the decision maker knows the state s_t when he makes the decision a_t. This MDP relaxation leads to poor lower bounds. We therefore use instead the larger RJT represented in Figure 10. In that RJT, $C_{a_t}^{\perp\!\!\perp} = \{s_t\}$, so that $G^{\perp\!\!\perp}$ is not anymore equal to $\overline{G}$ and the valid cuts enable one to enforce the independence of s_t and a_t given (s_{t-1}, a_{t-1}, o_t) for $t > 1$. Table 1 provides the numerical results on our instances. This example is harder to solve to optimality. SPU has worse performance as well on this example, and our formulations manage to improve the solution found by SPU. Once again, the valid cuts significantly reduce the linear relaxation gap and the solving time, even on large instances.

Conclusion

This paper introduced linear programming and MILP approaches for the MEU problem on IDs. The variables of the programs correspond, for the distributions induced by feasible policies, to the collection of vector of moments of the distribution on subsets of the variables that are associated to nodes of a new kind of junction tree, which we call an RJT. We have also proposed algorithms to build RJTs tailored to our linear and integer programs.

Soluble IDs are IDs whose MEU problem is easy, in the sense that it can be solved by the SPU algorithm. We showed that for soluble IDs the MEU problem can also be solved exactly via our linear programs. Furthermore, we characterized soluble IDs as the IDs for which there exists a junction tree such that the set of possible vector of moments on the nodes of the tree is convex for any parameterization of the ID.

Finally, we proposed a MILP approach to solve the MEU problem on nonsoluble IDs together with valid cuts. The bound provided by the linear relaxation is better than the bound that could be obtained using SPU on a soluble relaxation. Numerical experiments show that the bound is indeed better in practice.

Two elements limit the scale of the problems that can be dealt with using our approach. First, we use exact inference, which limits the applicability to models with small treewidth. Second, RJTs may contain clusters larger than those of standard junction trees. A possible way to overcome these limitations in future works would be to develop mathematical programming heuristics for IDs that use variational inference instead of exact inference.

Acknowledgments

The authors thank the reviewers for their thorough reading and all their useful comments and suggestions that helped them improve the paper.

Endnotes

[1] PIDs are a definition that we introduced and which is not introduced in Koller and Friedman (2009). We introduce it to distinguish properties due to the parameterization from properties due to the graph itself.

[2] If Alice did not want to play every day, we would also need to model her decisions. In that case, Bob and Alice would have different objectives and we would need to use a multiagent ID (Koller and Milch 2003). However, since Alice wants to play chess every day, her decisions do not need to be taken into account, and we can model the problem as an ID.

[3] See https://github.com/Victor2175/mathprog_influence_diagrams (accessed on June 10, 2020).

[4] The probabilistic graphical model community sometimes calls a *directed tree* what we call here a *rooted tree*, and a *polytree* what we call here a *directed tree*.

[5] The separators are often included in the definition of the junction tree and their associated moments τ_S in the definition of the marginal polytope. We do not include them in this work, because we do not need them in our mathematical programming formulations. Adding them would increase the size of the mathematical program and downgrade the performance of the solver.

[6] The concept of strong junction tree relies on the notion of *elimination ordering* for a given ID with perfect recall. The main difference is that a strong junction tree is a notion on an ID, where the set of decision vertices and their orders play a role, whereas RJTs rely on the underlying digraph. The notion of strong junction tree is obtained by replacing (ii) in the definition of an RJT by "given an elimination ordering, if (C_u, C_v) is an arc, there exists an ordering of C_v that respects the elimination ordering such that $C_u \cap C_v$ is before $C_v \backslash C_u$ in that ordering." An RJT is a strong junction tree. Conversely, a strong junction tree is not necessarily an RJT. Indeed, Jensen et al. (1994, figure 4) shows an example of strong junction where there is $v \in V$ such that $\mathrm{fa}(v) \subsetneq C_v$. Since strong junction trees is a notion on IDs and not on graphs, Theorem 2 has no natural generalization for strong junction trees.

[7] Although the particular RJT obtained by Algorithm 1 is a bucket tree, considering RJTs that are not bucket trees is also useful. Figure 10 shows an application where it is interesting to use an RJT that is not a bucket tree.

[8] These graphs are called *hypertrees* in the probabilistic graphical model literature.

[9] Relevance graphs are introduced in Section 6.

[10] We remind the reader that V^{r} is the set of utility vertices as introduced in Section 1.1.

[11] Indeed, the constraint $\mu \in \overline{\mathcal{P}}(G, \mathcal{X}, \mathfrak{p}, \mathcal{G})$ ensures that, for each $v \in V^{\mathrm{a}}$, μ_{C_v} is a distribution. The constraint $\mu_{C_v} = \delta_{v|\mathrm{pa}(v)}\, \mu_{\check{C}v}$ ensures that if $x_{\mathrm{pa}(v)}$ has nonzero probability according to that distribution, that is, $\sum_{x_{\check{C}v \backslash x_{\mathrm{pa}(v)}}} \mu_{\check{C}v}(x_{\check{C}v\backslash x_{\mathrm{pa}(v)}}, x_{\mathrm{pa}(v)}) > 0$, then $\delta_{v|\mathrm{pa}(v)}(\cdot|x_{\mathrm{pa}(v)})$ is a conditional probability.

[12] This type of property is well known in the literature on causality in graphical models, where the policies are viewed as *interventions* and some conditional probabilities are shown to be *invariant* under interventions: see, for example, Koller and Friedman (2009, definition 21.3, p. 1019) or Peters et al. (2017, remark 6.40, p. 113).

[13] In fact, if the graph is soluble and if the decision nodes are ordered in reverse topological order for the relevance graph, then SPU converges after exactly one pass over the nodes.

[14] When Nilsson and Höhle (2001) wrote their paper, the notion of soluble ID was still not known, and they used a weaker version. Their terminology is also different.

[15] The number of constraints defining polytopes $\overline{\mathfrak{Q}}^1$ and $\overline{\mathfrak{Q}}^b$ is $\sum_{v \in V^{\mathrm{s}}} |\mathcal{X}_{C_v}| + 3 \sum_{v\in V^{\mathrm{a}}} |\mathcal{X}_{C_v}| + \sum_{(C_u, C_v)\in\mathcal{A}} |\mathcal{X}_{C_u \cap C_v}|$, where $|\mathcal{X}_{C_v}| = \prod_{u\in C_v} |\mathcal{X}_u|$ for all v in V. If we use valid cuts, then the number of constraints of polytopes $\mathfrak{Q}^{\perp\!\!\perp,1}$ and $\mathfrak{Q}^{\perp\!\!\perp,b}$ is $2\sum_{v\in V^{\mathrm{s}}} |\mathcal{X}_{C_v}| + 4\sum_{v \in V^{\mathrm{a}}} |\mathcal{X}_{C_v}| + \sum_{(C_u,C_v)\in\mathcal{A}} |\mathcal{X}_{C_u\cap C_v}|$.

[16] This follows from the characterization of soluble IDs in Appendix D and the fact that $\vartheta_{\alpha_{t-1}} \not\perp_{G^\dagger} \mathrm{des}(a_t) | \mathrm{pa}(a_t)$ for all $t \in [T]$.

References

Adelman D, Mersereau AJ (2008) Relaxations of weakly coupled stochastic dynamic programs. *Oper. Res.* 56(3):712–727.

Antonucci A, Zaffalon M (2008) Decision-theoretic specification of credal networks: A unified language for uncertain modeling with sets of Bayesian networks. *Internat. J. Approximate Reasoning* 49(2):345–361.

Aras R, Dutech A (2010) An investigation into mathematical programming for finite horizon decentralized POMDPs. *J. Artificial Intelligence Res.* 37(1):329–396.

Bertsimas D, Mišić VV (2016) Decomposable Markov decision processes: A fluid optimization approach. *Oper. Res.* 64(6):1537–1555.

Bertsimas D, Niño-Mora J (2000) Restless bandits, linear programming relaxations, and a primal-dual index heuristic. *Oper. Res.* 48(1):80–90.

Bezanson J, Edelman A, Karpinski S, Shah VB (2017) Julia: A fresh approach to numerical computing. *SIAM Rev.* 59(1):65–98.

Cano A, Cano JE, Moral S (1994) Convex sets of probabilities propagation by simulated annealing. Globerson A, Silva R, eds. *Proc. 5th Internat. Conf. IPMU'94* (AUAI Press, Arlington, VA), 4–8.

Chandrasekaran V, Srebro N, Harsha P (2008) Complexity of inference in graphical models. *Proc. 24th Conf. Uncertainty Artificial Intelligence*, 70–78.

Cohen V, Parmentier A (2019) Two generalizations of Markov blankets. Preprint, submitted March 8, https://arxiv.org/abs/1903.03538.

de Campos CP, Cozman FG (2007) Inference in credal networks through integer programming. *Proc. 5th Internat. Sympos. Imprecise Probab. Theories Appl.*, 145–154.

de Campos CP, Ji Q (2012) Strategy selection in influence diagrams using imprecise probabilities. Preprint, submitted June 13, https://arxiv.org/abs/1206.3246.

De Farias DP, Van Roy B (2003) The linear programming approach to approximate dynamic programming. *Oper. Res.* 51(6):850–865.

Dechter R (2013) *Reasoning with Probabilistic and Deterministic Graphical Models: Exact Algorithms* (Morgan & Claypool Publishers, San Rafael, CA).

d'Epenoux F (1963) A probabilistic production and inventory problem. *Management Sci.* 10(1):98–108.

Dunning I, Huchette J, Lubin M (2017) JuMP: A modeling language for mathematical optimization. *SIAM Rev.* 59(2):295–320.

Eckles JE (1968) Optimum maintenance with incomplete information. *Oper. Res.* 16(5):1058–1067.

Hawkins JT (2003) A Lagrangian decomposition approach to weakly coupled dynamic optimization problems and its applications. Unpublished doctoral dissertation, Massachusetts Institute of Technology, Cambridge.

Howard RA, Matheson JE (1984) Influence diagrams. Howard RA, Matheson JE, eds. *Readings in the Principles and Practice of Decision Analysis* (Strategic Decision Systems, Menlo Park, CA), 719–762.

Howard RA, Matheson JE (2005) Influence diagrams. *Decision Anal.* 2(3):127–143.

Hurley B, O'Sullivan B, Allouche D, Katsirelos G, Schiex T, Zytnicki M, De Givry S (2016) Multi-language evaluation of exact solvers in graphical model discrete optimization. *Constraints* 21(3):413–434.

Jensen F, Jensen FV, Dittmer SL (1994) From influence diagrams to junction trees. *Proc. 10th Internat. Conf. Uncertainty Artificial Intelligence*, 367–373.

Kask K, Dechter R, Larrosa J, Dechter A (2005) Unifying tree decompositions for reasoning in graphical models. *Artificial Intelligence* 166(1):165–193.

Khaled A, Hansen EA, Yuan C (2013) Solving limited-memory influence diagrams using branch-and-bound search. Preprint, submitted September 26, https://arxiv.org/abs/1309.6839.

Koller D, Friedman N (2009) *Probabilistic Graphical Models: Principles and Techniques* (MIT Press, Cambridge, MA).

Koller D, Milch B (2003) Multi-agent influence diagrams for representing and solving games. *Games Econom. Behav.* 45(1):181–221.

Lauritzen SL, Nilsson D (2001) Representing and solving decision problems with limited information. *Management Sci.* 47(9):1235–1251.

Lee J, Ihler AT, Dechter R (2018) Join graph decomposition bounds for influence diagrams. Globerson A, Silva R, eds. *Proc. 34th Conf. Uncertainty Artificial Intelligence* (AUAI Press, Arlington, VA), 1053–1062.

Li Y, Yin B, Xi H (2011) Finding optimal memoryless policies of POMDPs under the expected average reward criterion. *Eur. J. Oper. Res.* 211(3):556–567.

Littman ML (1994a) Memoryless policies: Theoretical limitations and practical results. Cliff D, Husbands P, Meyer J-A, Wilson SW, eds. *Conf. Simulation Adaptive Behav.: From Animals to Animats 3* (MIT Press, Cambridge, MA), 238–245.

Littman ML (1994b) The witness algorithm: Solving partially observable Markov decision processes. Technical report.

Liu Q (2014) Reasoning and decisions in probabilistic graphical models–a unified framework. Unpublished doctoral dissertation, University of California, Irvine.

Maua DD (2016) Equivalences between maximum a posteriori inference in Bayesian networks and maximum expected utility computation in influence diagrams. *Internat. J. Approximate Reasoning* 68:211–229.

Mauá DD, Cozman FG (2016) Fast local search methods for solving limited memory influence diagrams. *Internat. J. Approx. Reason.* 68:230–245.

Mauá DD, de Campos C (2011) Solving decision problems with limited information. *Adv. Neural Inform. Processing Systems* 24:603–611.

Mauá DD, de Campos CP, Zaffalon M (2012a) The complexity of approximately solving influence diagrams. Globerson A, Silva R, eds. *Proc. 28th Conf. Uncertainty Artificial Intelligence* (AUAI Press, Arlington, VA), 604–613.

Mauá DD, de Campos CP, Zaffalon M (2012b) Solving limited memory influence diagrams. *J. Artificial Intelligence Res.* 44:97–140.

Mauá DD, De Campos CP, Zaffalon M (2013) On the complexity of solving polytree-shaped limited memory influence diagrams with binary variables. *Artificial Intelligence* 205:30–38.

McCormick GP (1976) Computability of global solutions to factorable nonconvex programs: Part I – convex underestimating problems. *Math. Programming* 10(1):147–175.

Nilsson D, Höhle M (2001) Computing bounds on expected utilities for optimal policies based on limited information. Technical report, Danish Informatics Network in the Agriculture Sciences.

Papadimitriou CH, Tsitsiklis JN (1987) The complexity of Markov decision processes. *Math. Oper. Res.* 12(3):441–450.

Peters J, Janzing D, Schölkopf B (2017) *Elements of Causal Inference: Foundations and Learning Algorithms* (MIT Press, Cambridge, MA).

Poh KL, Horvitz E (1996) *A Graph-Theoretic Analysis of Information Value* (UAI) (Morgan Kaufman Publishers, San Francisco).

Powell WB (2014) Clearing the jungle of stochastic optimization. Newman A, Leung J, eds. *Bridging Data and Decisions* (NFORMS, Catonsville, MD), 109–137.

Robertson N, Seymour PD (1983) Graph minors. I. Excluding a forest. *J. Combin. Theory Ser. B* 35(1):39–61.

Ross S, Pineau J, Paquet S, Chaib-draa B (2008) Online planning algorithms for POMDPs. *J. Artificial Intelligence Res.* 32(1):663–704.

Scheffler P (1990) A linear algorithm for the pathwidth of trees. *Topics in Combinatorics and Graph Theory* (Springer), 613–620.

Schrijver A (2003) *Combinatorial Optimization: Polyhedra and Efficiency* (Springer, Berlin).

Shachter RD (1986) Evaluating influence diagrams. *Oper. Res.* 34(6):871–882.

Shani G, Pineau J, Kaplow R (2013) A survey of point-based POMDP solvers. *Autonomous Agents Multi-Agent Systems* 27(1):1–51.

Shenoy PP (1992) Valuation-based systems for Bayesian decision analysis. *Oper. Res.* 40(3):463–484.

Smallwood RD, Sondik EJ (1973) The optimal control of partially observable Markov processes over a finite horizon. *Oper. Res.* 21(5):1071–1088.

Sontag D, Globerson A, Jaakkola T (2011) Introduction to dual decomposition for inference. *Optimization for Machine Learning*, 219–254.

Wainwright MJ, Jordan MI (2008) Graphical models, exponential families, and variational inference. *Foundations Trends Machine Learn.* 1(1–2):1–305.

Yuan C, Wu X, Hansen E (2010) Solving multistage influence diagrams using branch-and-bound search. Globerson A, Silva R, eds. *Proc. 26th Conf. Uncertainty Artificial Intelligence* (AUAI Press, Arlington, VA), 691–700.