Journal of Mathematical Psychology 57 (2013) 117–133

Contents lists available at ScienceDirect

Journal of Mathematical Psychology

journal homepage: www.elsevier.com/locate/jmp

A computational-level explanation of the speed of goal inference

Mark Blokpoel [a,*], Johan Kwisthout [a,b], Theo P. van der Weide [b], Todd Wareham [c], Iris van Rooij [a]

[a] Radboud University, Donders Institute for Brain, Cognition and Behaviour, Nijmegen, The Netherlands
[b] Radboud University, Institute for Computing and Information Sciences, Nijmegen, The Netherlands
[c] Department of Computer Science, Memorial University of Newfoundland, Canada

HIGHLIGHTS

- We reflect on the intractability of Bayesian models of goal inference.
- We explain the use of a complexity-theoretic methodology.
- We present computational-level explanations of the speed of goal inferences.

ARTICLE INFO

Article history:
Received 10 July 2012
Received in revised form
29 May 2013
Available online 2 July 2013

Keywords:
Goal inference
Abduction
Inverse planning
Computational complexity
Intractability
NP-hard
Fixed-parameter tractability

ABSTRACT

The ability to understand the goals that drive another person's actions is an important social and cognitive skill. This is no trivial task, because any given action may in principle be explained by different possible goals (e.g., one may wave ones arm to hail a cab or to swat a mosquito). To select which goal best explains an observed action is a form of abduction. To explain how people perform such abductive inferences, Baker, Tenenbaum, and Saxe (2007) proposed a computational-level theory that formalizes goal inference as Bayesian inverse planning (BIP). It is known that general Bayesian inference – be it exact or approximate – is computationally intractable (NP-hard). As the time required for computationally intractable computations grows excessively fast when scaled from toy domains to the real world, it seems that such models cannot explain how humans can perform Bayesian inferences quickly in real world situations. In this paper we investigate how the BIP model can nevertheless explain how people are able to make goal inferences quickly. The approach that we propose builds on taking situational constraints explicitly into account in the computational-level model. We present a methodology for identifying situational constraints that render the model tractable. We discuss the implications of our findings and reflect on how the methodology can be applied to alternative models of goal inference and Bayesian models in general.

1. Introduction

An important aspect of human sociality is our ability to understand the actions of others as being goal-directed. We seem to be able to often effortlessly understand which goals motivate the behavior of others that we observe. The apparent ease and speed with which humans are able to perform goal inference stands in sharp contrast to the computational challenges that such an inference seems to pose. Several authors (Baker et al., 2007; Charniak & Goldman, 1993; Uithol, van Rooij, Bekkering, & Haselager, 2011; van Rooij, Haselager, & Bekkering, 2008) have suggested that goal inference is a form of 'inference to the best explanation', also known as abduction, which involves a form of reasoning from observations (here, actions) to hypothesized causes that explain those observations (here, goals). Abduction is notorious in both philosophy and artificial intelligence for its intractability (Abdelbar & Hedetniemi, 1998; Fodor, 2001; Haselager, van Dijk, & van Rooij, 2008; Pylyshyn, 1987), and formal proofs are available that the computations involved in abduction can require a combinatorially explosive number of basic computational steps (e.g., Bylander, Allemang, Tanner, & Josephson, 1991; Shimony, 1994). This observation raises the question of how one can explain the speed of goal inference given that it is a form of abduction. In this paper we present a methodology for addressing this question, and illustrate its use for a particular model of goal inference.

In line with a long tradition of explaining goal inference (Baker et al., 2007; Baldwin & Baird, 2001; Charniak & Goldman, 1993; Cuijpers, van Schie, Koppen, Erlhagen, & Bekkering, 2006; Dennett,

* Corresponding author.
E-mail address: m.blokpoel@donders.ru.nl (M. Blokpoel).

0022-2496/$ – see front matter
http://dx.doi.org/10.1016/j.jmp.2013.05.006

1987; Hassin, Aarts, & Ferguson, 2005; Király, Jovanovic, & Prinz, 2003; van Rooij, Haselager et al., 2008) have proposed that goal inference can be seen as a form of *inverse planning*, similar to how vision can be seen as a form of inverse graphics (e.g., Pizlo, 2008). Baker et al. go beyond existing psychological approaches (see, e.g., Csibra & Gergely, 2007; Csibra, Gergely, Biró, Koós, & Brockbank, 1999; Gergely, Nádasdy, Csibra, & Biró, 1995) by providing a precise formalization of inverse planning in the form of a Bayesian inference model.

According to the Bayesian Inverse Planning (BIP) model, observers assume that actors are 'rational' in the sense that they tend to adopt those actions that best achieve their goals. Given the assumption of rationality, and (probabilistic) knowledge of the world and how actions are affected by it, the probability that an agent performs an action given its goals within a certain context can be defined as the following probabilistic dependency:

$$\Pr(action \mid goal, context). \tag{1}$$

Here, context can be any relevant information. Conversely, it is assumed that an observer interested in understanding why an actor acts the way she does infers which goals best explain (i.e., are most likely given) her actions and the context. To describe how an observer infers the most likely goal, the conditional probability that described planning Eq. (1) can be inverted using Bayes' rule:

$$\Pr(goal \mid action, context) \propto \Pr(action \mid goal, context)\Pr(goal \mid context). \tag{2}$$

Of all the possible goals that an observer can (or does) entertain, the goal that maximizes the probability in Eq. (2) is taken to be the one that *best explains* why the observed action was performed in this context and is the goal that is inferred.[1] The BIP model has been tested in several experiments, and Baker, Saxe, and Tenenbaum (2009); Baker et al. (2007) observed that it can account for the dynamics of goal inferences made by human participants in several different experimental settings.

1.1. Dealing with intractability

Although the BIP model is able to describe human behavior under lab conditions, this does not yet mean that it *explains* this behavior. Explanation requires more than fit alone (Cummins, 2000). For instance, for a computational-level explanation of a cognitive ability to explain the functioning of that ability in everyday situations, the fit found in the lab should be able to scale to situations outside the lab. The BIP model belongs to the class of (rational) Bayesian inference models, which are generally taken to be intractable by proponents and opponents alike (e.g., Chater, Tenenbaum, & Yuille, 2006; Gigerenzer, Hoffrage, & Goldstein, 2008). This raises the question of whether or not the computations that the model postulates can scale to situations of everyday complexity and thus if the model is a plausible explanation of everyday goal inference. As Gigerenzer and colleagues put it:

> The computations postulated by a model of cognition need to be tractable in the real world in which people live, not only in the small world of an experiment with only a few cues. This eliminates NP-hard models that lead to computational explosion, such as probabilistic inference using Bayesian belief networks … including its approximations (Gigerenzer et al., 2008, p. 236).

Gigerenzer and colleagues rightly point out that intractable models cannot plausibly scale and hence are explanatorily unsatisfactory as models of cognition. However, eliminating all Bayesian models of cognition as suggested by Gigerenzer and colleagues seems unnecessary. It is true that inferences on Bayesian networks can be intractable if no constraints are imposed on the networks, but the same computations may become tractable under the right set of constraints (Kwisthout, 2011). More generally, even if a modeling framework can include intractable models then that does not imply all models in the framework are necessarily intractable. For this reason, rejecting a whole modeling framework based on intractability results for specific models seems unjustified (see also van Rooij, 2008, pp. 951–952).

In contrast to the framework rejection response, a more common response to intractability by proponents of Bayesian models is to postulate inexact algorithms as process explanations (Chater et al., 2006; Sanborn, Griffiths, & Navarro, 2010; van Rooij, Wright, & Wareham, 2012). The idea behind this approach seems to be that, given Bayesian computations (at Marr's (1982) computational level) are intractable, human minds/brains at best can approximate such computations using inexact algorithms (Marr's algorithmic level). This presupposes that approximating intractable Bayesian computations is tractable. Yet, it is known that many such intractable computations are in fact also intractable to approximate (see also Kwisthout, Wareham, & van Rooij, 2011; van Rooij & Wareham, 2012).[2]

In our view, a key to understanding the computational feasibility of a Bayesian (or any cognitive) model lies in studying domain-specific constraints that may hold in the model's domain of application (e.g., action understanding or vision). It is well known in computational complexity theory that an intractable function $f : I \rightarrow O$ can be tractable for a restricted input domain $I' \subset I$. This property can be used to explain why Bayesian computations which are intractable for unrestricted input may nevertheless be able to explain the speed of human inferences, namely, under the assumption that the latter is operating on a proper subset of all the inputs that the general model encompasses. If the computational-level theory is adapted to include the same input constraints that are operational for ecological inputs for humans, then the model should inherit the tractability that holds for human inferences. In this perspective, the challenge of explaining the speed of human inferences thus lies in identifying those input constraints.[3]

In this paper we illustrate the use of a complexity-theoretic methodology for identifying input constraints that render the Bayesian Inverse Planning (BIP) model of goal inference tractable. Our goal is not only to contribute to our understanding of what makes speedy goal inferences possible under the BIP model, but also to illustrate our methodology in sufficient detail so that others can adopt it to analyze other computational-level models, whether Bayesian or otherwise. For this reason we try to make our paper as self-contained as possible. For instance, we include primers on the two mathematical domains relevant for our analyses, namely, Bayesian modeling and computational complexity theory. Further, we make our mathematical proofs accessible to a wide readership by informally describing the core ideas behind them and by elucidating them with graphical illustrations. We will show that

[1] This definition of 'best explanation' is standard in the field of Bayesian abduction (Kwisthout, 2011). We note, however, that alternative definitions have also been proposed (Glass, 2007; Kwisthout, 2010). Some complexity results may generalize to such alternative definitions as well, though some may not. Verifying which complexity results generalize requires further analyses.

[2] This is not to say that there are no intractable computations that can be tractably approximated (see e.g., Kwisthout & van Rooij, 2013), but this seems rare (Arora, 1998) and does not generally hold for unconstrained Bayesian computations (Kwisthout et al., 2011).

[3] Besides framework rejection and inexact algorithms, other ways of dealing with intractability have been proposed (including attempts to deny that it is a real issue). For a discussion of the limitations of these alternative approaches we refer the interested reader to van Rooij (2008), van Rooij et al. (2012) and Kwisthout et al. (2011).

by using the complexity-theoretic methodology one can *explain* when and why speedy goal inference is possible under the BIP model and that this explanation is situated at the *computational-level*. This is an important finding in that it stands in contrast to the standard assumption that intractability at the computational-level can only be remedied by introducing inexact algorithms as (approximate) algorithmic-level explanations (Chater et al., 2006; Thagard & Verbeurgt, 1998; van Rooij et al., 2012).

1.2. Overview

We first introduce basic concepts from Bayesian modeling in Section 2, which are used to formally define the Bayesian Inverse Planning model in Section 3. In Section 4 we introduce basic concepts from computational complexity theory and explain how they lay the foundation for a methodology for identifying constraints that can render intractable models tractable. We apply this methodology to analyze the BIP model in Section 4. Specifically, we will present proofs that the BIP model is intractable (NP-hard) when its input domain is unconstrained, yet tractable under a set of constraints on its input domain. We close, in Section 6, by discussing the implications of our results for the study of goal inference in particular and for dealing with the issue of intractability in cognitive science and Bayesian modeling in general.

2. Concepts from Bayesian modeling

In this section we review concepts and notation from Bayesian modeling that are used in the remainder of this paper. Readers familiar with Bayes' rule, Bayesian computations (such as Most Probable Explanation), the math underlying Bayesian network diagrams and their computational complexity can skip this section without loss of information. Readers interested in a more detailed account of Bayesian models are referred to textbooks such as Pearl (1988) or Jensen and Nielsen (2007).

2.1. Notation and definitions

We denote variables with capital letters, whereas small letters are used to denote values (e.g., A and a, such that a denotes the value of A). The domain of a variable is denoted by the function $\Omega(.)$, which returns the set of all possible values of a variable. A bold capital letter, such as $\mathbf{V}$, represents a set of variables and a bold small letter, such as $\mathbf{v}$, denotes a joint value assignment for a set of variables. More formally, a joint value assignment such as $\mathbf{v}$ to a set of n variables $\mathbf{V}$ is an n-tuple that assigns a value $v \in \Omega(V)$ to each variable $V \in \mathbf{V}$. Using the definition of a joint value assignment, the function $\Omega(.)$ can also be applied to sets of variables. It then returns a set of all possible joint value assignments for those variables. Additionally, the function $\pi(.)$ returns the set of parents for a particular vertex in a directed acyclic graph (DAG) and is defined as $\pi(X) = \{V' | (V', X) \in A\}$, where A is a set of arcs in that DAG. We abbreviate $A = a$ to a or $\mathbf{A} = \mathbf{a}$ to $\mathbf{a}$ where no ambiguity is possible and we sometimes use ',' rather than '$\wedge$' to abbreviate long propositions.

Furthermore, we introduce the following four concepts:

- A *joint distribution* defines all the probabilities for all possible value assignments to all variables in a Bayesian network.
- A *joint probability* is the probability of a specific joint value assignment to all variables in a Bayesian network.
- A *marginal distribution* defines all the probabilities for all possible value assignments to a strict subset of variables in a Bayesian network.
- A *marginal probability* is the probability of a specific joint value assignment to a proper subset of variables in a Bayesian network.

2.2. Bayesian probability

Bayesian probabilities denote the 'degree of belief' in the truth of propositions subject to the constraints of probability theory. For example, the probability that 'the red cup is red' is 1; but the probability that 'the patient has influenza' can range from 0 to 1. Combinations of these propositions lead to different beliefs. The probability that 'the patient has influenza *and* the patient has a fever' is different from (but possibly related to) the probability of the individual propositions. For the purposes of this paper the function Pr returns the probability of a proposition. We formalize propositions using Boolean algebra ξ. The atomic element of a proposition is the equation $A = a$, which returns true if variable A has value a and false otherwise. The equality function for sets and joint value assignments $\mathbf{A} = \mathbf{a}$ is defined similarly, i.e., it returns true if every variable in $\mathbf{A}$ has the associated value from the joint value assignment $\mathbf{a}$ or false otherwise. The Boolean algebra ξ is a six-tuple $\langle Q, \wedge, \vee, \neg, \top, \bot \rangle$, where Q is a set of atomic elements. We can now define a *joint probability distribution* as a function $\Pr : \xi \rightarrow [0, 1]$ that for any possible proposition in ξ returns a belief (or probability) value ranging from 0 to 1 (inclusive).

A joint probability distribution has the following axioms:

1. The probability of any proposition lies between 0 and 1 (inclusive): $0 \leq \Pr(a) \leq 1$, for all $a \in \xi$;
2. If any two propositions a and b are logically equivalent then their probability is the same: $a \equiv b \Rightarrow \Pr(a) = \Pr(b)$;
3. The belief in a true statement is 1: $\Pr(\top) = 1$;
4. The belief in a false statement is 0: $\Pr(\bot) = 0$;
5. For any two propositions $a, b \in \xi$ that are exclusive (i.e., $a \wedge b \equiv \bot$) it holds that $\Pr(a \vee b) = \Pr(a) + \Pr(b)$.

From these axioms it can be shown that if two propositions $a, b \in \xi$ are not exclusive (i.e., $a \wedge b \not\equiv \bot$) then it holds that $\Pr(a \vee b) = \Pr(a) + \Pr(b) - \Pr(a \wedge b)$.

2.3. Bayesian networks

An elegant and common way to represent joint probability distributions is by using *Bayesian networks* (BNs) (Ghahramani, 1998; Jensen & Nielsen, 2007; Pearl, 1988). In such networks, probabilistic relations between variables (represented by nodes) are defined using conditional probabilities. The conditional probability of a given b is defined as $\Pr(a \mid b) = \Pr(a \wedge b) / \Pr(b)$. Dependencies between variables are graphically depicted using directed arcs between the corresponding nodes in the network, where an arc from a to b denotes that the probability distribution of b is conditioned on a.

In a BN $\mathcal{B} = \langle G = (\mathbf{V}, \mathbf{A}), \Gamma \rangle$ variables make up the vertices $\mathbf{V}$ of the directed acyclic graph (DAG) G and dependencies between variables are represented as arcs $\mathbf{A}$. The set $\Gamma = \{\Pr_X | X \in \mathbf{V}\}$ contains all conditional probability distributions of all variables in $\mathbf{V}$. Here, $\Pr_X$ is a conditional probability distribution for X. $\Pr_X$ is a function that, given a value x for X and a joint value assignment $\mathbf{y}$ to the parents $\mathbf{Y}$ of X, returns the probability $\Pr(X = x \mid \mathbf{Y} = \mathbf{y})$. To illustrate how Bayesian networks can be used to capture the probabilistic relationships within a real-world domain and how Bayesian inference can be used to reason with this information we look at a small example network reflecting fictitious medical knowledge. The BN in Fig. 1 is a model of the dependencies between two diseases and their symptoms. There are no direct dependencies between `influenza` and `pneumonia`, but they may become dependent on each other given evidence for one or the other. For example, if a patient has a fever, then the observation that she has influenza actually decreases the probability of pneumonia, as influenza by itself is sufficient to explain the fever. An obvious dependency is related to the effect

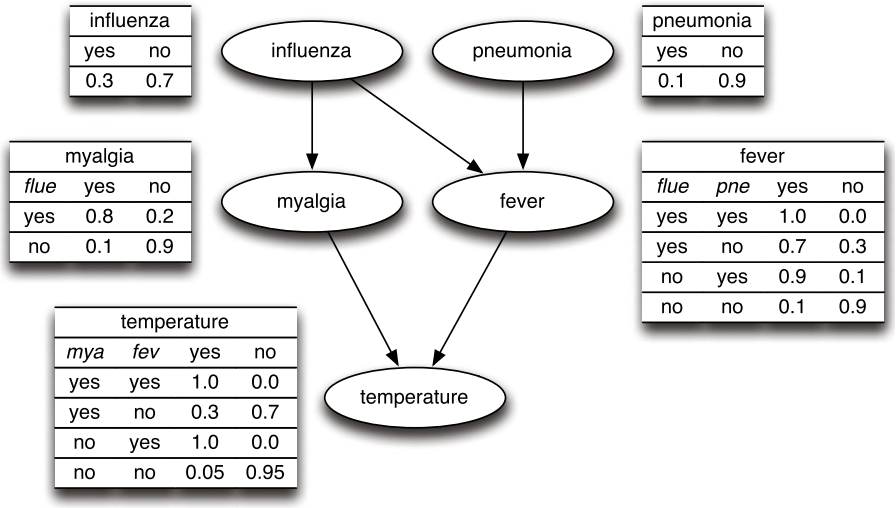

Fig. 1. A classic example BN describing the dependencies and conditional probabilities for two diseases and their symptoms. When no ambiguity can occur we use the first letter of the variable to denote that variable (e.g., I for influenza).

of having (or not having) influenza on whether or not a patient has myalgia or a fever. These dependencies are represented by the corresponding arcs between the variables. Given this network and the conditional probabilities (depicted in the tables in the same figure) one can compute the probability that a patient has influenza or pneumonia once particular variables are observed. This computation is called Bayesian inference and can be done using a combination of the following properties and rules:

1. *Chain rule*, a joint probability distribution can be computed using conditional probabilities:
   $\Pr(x_1, \ldots, x_n) = \Pr(x_n \mid x_1, \ldots, x_{n-1}) \cdot \ldots \cdot \Pr(x_2 \mid x_1) \cdot \Pr(x_1)$
2. *Marginalization*, joint probability distributions sum up to marginal distributions: $\Pr(\mathbf{y}) = \sum_{x_i \in \Omega(X)} \Pr(\mathbf{y} \wedge x_i)$
3. *Conditioning*, a marginal distribution can be computed by summing over all conditional probabilities:
   $\Pr(\mathbf{y}) = \sum_{x_i \in \Omega(X)} \Pr(\mathbf{y} \mid x_i) \cdot \Pr(x_i)$
4. *Bayes' rule*, conditional probabilities can be inversed:
   $\Pr(\mathbf{y} \mid \mathbf{x}) = \frac{\Pr(\mathbf{x}|\mathbf{y}) \cdot \Pr(\mathbf{y})}{\Pr(\mathbf{x})}$, assuming $\Pr(\mathbf{y}), \Pr(\mathbf{x}) > 0$.

2.4. Most probable explanation

We now consider the example BN from Fig. 1 to demonstrate how Bayesian inference can be used to reason from observations (e.g., a patient has a high temperature) to the underlying most probable causes. One of the possibilities is that the patient has influenza, but is that the most probable explanation of the observations? We can compute the posterior probability of this explanation by computing the probability that 'the patient has influenza' given that 'the patient has a high temperature' (i.e., $\Pr(I = \mathit{true} \mid T = \mathit{true})$ which we abbreviate to $\Pr(i \mid t)$). By applying Bayes' rule we get:

$$\Pr(i \mid t) = \frac{\Pr(t \mid i) \cdot \Pr(i)}{\Pr(t)}$$

Then by applying the marginalization and conditioning rules we get:

$$\frac{\left(\sum_{M,F,P} \Pr(t|M,F) \cdot \Pr(M|i) \cdot \Pr(F|i,P)\right) \cdot \Pr(i)}{\Pr(t)} =$$

$$\frac{\left(\sum_{M,F,P} \Pr(t|M,F) \cdot \Pr(M|i) \cdot \Pr(F|i,P)\right) \cdot \Pr(i)}{\sum_{M,F,I,P} \Pr(t|M,F) \cdot \Pr(M|I) \cdot \Pr(F|I,P) \cdot \Pr(I) \cdot \Pr(P)} =$$

$$\frac{0.1664}{0.2899} \approx 0.57.$$

This formula illustrates that to calculate the posterior probability of a variable, given certain observations, one has to sum over all possible combinations of the values of other variables. This can potentially require vast amounts of computational resources when the number of variables and dependencies increase.

To find out if 'the patient has influenza' is the most probable explanation we would have to figure out whether the posterior probability of $\Pr(i \mid t)$ is higher than all potential causes (e.g., 'the patient has pneumonia'). This computational problem is known as Most Probable Explanation (MPE). It consists of finding the most probable joint value assignment for a set of variables $\mathbf{M}$ given evidence (i.e., a joint value assignment) $\mathbf{e}$ to $\mathbf{E}$, where $\mathbf{M} \cup \mathbf{E} = \mathbf{V}$ and $\mathbf{M} \cap \mathbf{E} = \varnothing$. For the medical example, the possible explanations could be $\mathbf{M}$ = {'influenza', 'pneumonia', 'myalgia'} and the observed evidence would then be $\mathbf{E}$ = {'fever', 'temperature'}. The Bayesian models of goal inference in this paper are based on the following formal definition of MPE:

Most Probable Explanation (MPE)
Input: A Bayesian network $\mathcal{B} = \langle \mathbf{G} = (\mathbf{V}, \mathbf{A}), \Gamma \rangle$, where $\mathbf{V}$ is partitioned into a set of evidence nodes $\mathbf{E}$ with a joint value assignment $\mathbf{e}$ and an explanation set $\mathbf{M}$.
Output: What is the most probable joint value assignment $\mathbf{m}$ to the nodes in $\mathbf{M}$ given evidence $\mathbf{e}$?

The computational complexity of this input–output mapping is well studied (for an overview see Kwisthout, 2011). In its general form it is computationally intractable (as defined in Section 4), partly due to the combinatorial explosion illustrated in the example computation.

3. Computational-level model of Bayesian inverse planning

Using the Bayesian formalism from the previous section we continue by formalizing Bayesian inverse planning at the computational level (Marr, 1982). In the first half of this section we introduce the concept of planning and inverse planning using an

example, and we show how this theory can be conceptualized in a formal model. The second half of this section is used to introduce a formal definition of the Bayesian inverse planning model that will be the subject of our investigation. The formal Bayesian inverse planning model in this paper is a more general version of the simplest model proposed by Baker et al. (2009, 2007) (what those authors refer to as M1), in the sense that it can accommodate for multiple (simultaneous) goal attributions. We chose to extend the simplest model, because it greatly simplifies the mathematical analysis (see Section 5) and, importantly, the results of that analysis also hold for analogous extensions of the more complex models proposed by Baker et al. (what those authors refer to as M2 and M3).[4]

3.1. An example of Bayesian inverse planning

Consider the following example. Imagine a mother and her son, sitting in the same room, when she hears his stomach rumble. She sees her son get up, walk to the kitchen and start searching for something. At first he finds a sour apple, which he discards in search of something else. Then the mother sees her son finding a delicious candy bar. When he starts to eat it she realizes her son is trying to still his hunger and at the same time wants to eat something sweet. In this scenario, the son goes through a process of *planning*, choosing his actions to achieve his goals. The mother observes the actions of her son and based on her observations infers the goals she thinks her son is trying to achieve. This process is called *goal inference*.

Baker et al. (2009) characterize goal inference as inverse probabilistic planning. They assume that observers assume that agents act 'rational' in the sense that their behavior is such that they best achieve their goals. Here 'best' may, for instance, be defined in terms of (expected or believed) efficiency of a set of actions for achieving a given (combination) of goal(s), which Baker et al. (2009) modeled as a Markov decision process. Say that, in our example, the son only has one goal 'satisfying hunger'. Then his behavior 'searching, finding sour apple, continue searching, finding candy bar, eat' achieves this goal, but it is less rational then the behavior 'searching, finding sour apple, eat' if one defines rationality in terms of efficiency. Be that as it may, the son's behavior can be rational for a different goal, e.g., if the son has two simultaneous goals 'to satisfy hunger' *and* 'to taste sweet'. Given such rationality an observer, in this case the mother, will more likely attribute to the son the two simultaneous goals rather than the one single goal.

The Bayesian inverse planning (BIP) model does not commit to any particular representation, i.e., aside from the requirement that the values stored in the variables are discrete the precise representations of these values and variables are unspecified. We can use the 'mother–son' example to illustrate how the BIP model can capture the mother's goal inference. To do this we first need to represent the son's behavior. This is modeled by a sequence of states **S** and actions **A**. As no particular representation is required by the model, we assume the labels for actions and states: `search`, `eat`, `stomach rumbles`, `finds sour apple`, `finds candy-bar` and `happily eating the bar`.[5]

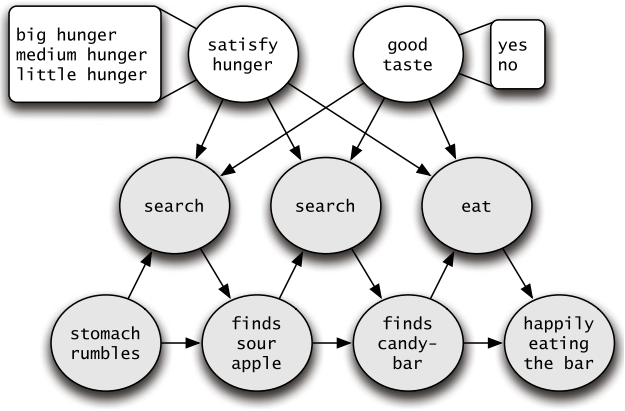

Fig. 2. A possible Bayesian network that models the Bayesian inverse planning performed by the mother in the example in Section 3.1. Shaded nodes represent observed variables, i.e., observed actions (e.g., search) and states (e.g., stomach rumbles), while white nodes represent the goal variables for which the mother has to infer the most likely values. Possible values are represented by the rounded rectangles.

At each moment in time the current state depends on the previous state and the action taken in that previous state. The action at a given moment in time depends on the current state and on the actor's goals. Goals are represented by goal variables. In our example the mother assumes her son's behavior is guided by two goals `satisfy hunger` and `taste sweet`. Based on these variables and dependencies we can construct a Bayesian network (see Fig. 2).

Once such a network including the conditional probabilities for all dependencies is established, one can characterize goal inference, as performed by the mother, as the most likely value assignment for the goal variables (i.e., Most Probable Explanation or MPE, see Section 2). Given the possible values in Fig. 2, this can be any combination consisting of one of {`big hunger`, `medium hunger`, `little hunger`} and one of {`yes`, `no`}.

3.2. A computational-level model of Bayesian inverse planning

The computational-level model we present in this section includes the simplest model (M1) proposed by Baker et al. (2009, 2007) as a special case, because it extends the model such that it can also account for people's ability to infer multiple goals simultaneously as illustrated by the 'mother–son' example. The model captures goal inference as a form of abduction, specifically as a special case of MPE. Given that our model is a computational-level model, it does not commit to a specific hypothesis about how the modeled goal inference is computed by humans. This could be done by computing probabilities for all possible goal assignments, but this is not necessarily the case. We characterize the extended Bayesian inverse planning model as the following informal input–output mapping:

Bayesian Inverse Planning (informal)

Input: A representation of the probabilistic dependencies between actions, goals and states (i.e., Pr(*actions* | *goals*, *states*) and Pr(*goals* | *state*)) and how these dependencies change over time, and a sequence of observed actions and world states.

Output: A combination of goals that best explains the sequence of actions and world states against the background of the probabilistic dependencies between actions, goals and world states and how these dependencies change over time (i.e., the goals that maximize Pr(*goals* | *actions*, *states*)).

[4] In Blokpoel, Kwisthout, van der Weide, and van Rooij (2010) we analyzed an even more general model of goal inference that had all models (M1, M2 and M3) as special cases. In that paper, was shown that the same (in)tractability results presented in Section 5 of this paper for our extension of M1 also hold for analogous extensions of M2 and M3.

[5] The fact that the model does not commit to particular representations but only to a particular structure (i.e., the (in)dependencies) means that it is consistent with any representation that is compatible with that structure.

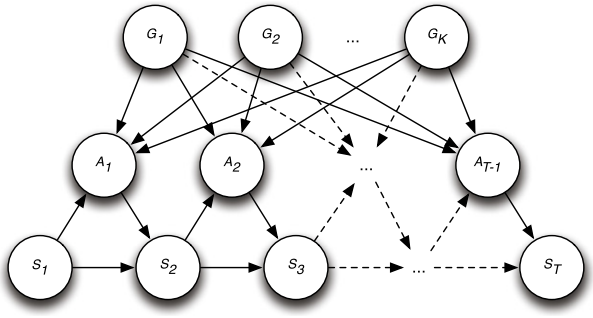

Fig. 3. The dynamic BN that underlies the BIP model. $\mathbf{S} = \{S_1, \ldots, S_T\}$ represent states, $\mathbf{A} = \{A_1, \ldots, A_{T-1}\}$ represent actions and $\mathbf{G} = \{G_1, \ldots, G_K\}$ represents the set of goals.

We now work towards formalizing the above BIP model. The input of the formal model will consist of three types of variables: *states* **S**, *actions* **A** and *goals* **G**. To represent the probabilistic dependencies between these variables and how these change over time we will use a special type of Bayesian network called a dynamic Bayesian network. A dynamic BN is a BN with a restricted structure that is based upon a discretized concept of time. At each time step t a dynamic BN contains a so called *slice*—a sub-BN—that represents the situation at time t. To model changes to the world over time, each slice is related to (i.e., is probabilistically dependent on) the preceding slice. In addition to slices, a dynamic BN can contain variables that are constant over time. Slices can be dependent on these 'global' variables. The goal variables **G** in the BIP model do not change over time and are hence global variables. For each time t, the slice corresponding to t consists of a state variable S_t and an action variable A_t. The action variable represents what action is observed at time t and depends on the state variable S_t and the actor's goals **G**. The state variable S_t depends both on S_{t-1} and A_{t-1} in the previous slice. See Fig. 3 for a graphical representation of the resulting dynamic Bayesian network.

There are two ways one can use this dynamic BN. Firstly, one can model planning: Given a joint value assignment **g** to the goals and the initial state s_1, determine the most probable joint value assignment to future actions and states **a**, **s** (i.e., find those actions that maximize Pr(*actions* | *goals*, *states*)). Secondly, as Baker et al. (2009, 2007) proposed, one can model action understanding by using Bayes' rule to invert the direction of the inference to infer an actor's goals given observed behavior. In this sense goal inference is modeled as the computation of the most likely joint value assignment **g** for the set of goals, given a joint value assignment **a**, **s** to states and actions.[6]

We can now formally define Bayesian Inverse Planning with multiple goals as the following input–output mapping:

Bayesian Inverse Planning (formal)
Input: A dynamic Bayesian network $\mathcal{B} = (\mathbf{G}_{\mathcal{B}}, \Gamma_{\mathcal{B}})$, where $\mathbf{G}_{\mathcal{B}}$ is a directed acyclic graph $\mathbf{G}_{\mathcal{B}} = (\mathbf{V}_{\mathcal{B}}, \mathbf{A}_{\mathcal{B}})$. The structure of $\mathbf{G}_{\mathcal{B}}$ is limited to a sequence of *states* **S** and *actions* **A** of length $T \geq 2$ and a non-empty set of K goal variables **G**, such that:

- any action A_t at time t depends on the state S_t at that time;
- any state S_t at time t depends on the previous state S_{t-1} and action A_{t-1};
- any action A can depend on all goal variables (depicted graphically by arcs from all goal variables to A).

Furthermore the input consists of a joint value assignment $\mathbf{s} \cup \mathbf{a}$ to all state and action variables.
Output: A joint value assignment **g** to all goal variables in **G** such that $\Pr(\mathbf{g} \mid \mathbf{s} \cup \mathbf{a})$ is maximized.

4. Concepts from computational complexity

Given that the formalization of Bayesian inverse planning in the previous section is at the computational level, we are now in a position to investigate constraints on the input domain of the theory that can or cannot render goal inference under the model tractable. To this end, we use concepts and proof techniques from computational complexity theory (Downey & Fellows, 1999; Garey & Johnson, 1979). In this section we review those concepts and notation from computational complexity that are used in the remainder of this paper. Readers familiar with concepts such as complexity classes (P, NP, FPT and W[1]), Big-Oh notation, fixed-parameter (in)tractability, and polynomial-time and parameterized reductions can skip this section without loss of information.

4.1. Notation and definitions

In the fields of computer science, and computational complexity in particular, the term *(computational) problem* is traditionally used to refer to a particular input–output mapping. In cognitive psychology, computational-level models are also input–output mappings. Therefore, in the discourse of this paper, the terms *computational problem* and *model* refer to the same mathematical object. For example, problem Π or model M are both defined by their respective input–output mappings:

$$\Pi : I_\Pi \rightarrow O_\Pi$$
$$M : I_M \rightarrow O_M.$$

There are several types of problems. Given some function $val : O \rightarrow \mathbb{N}$, an *optimization problem* returns the argument that maximizes or minimizes the value of that function val (depending on whether the optimization problem is a maximization or minimization problem, respectively). For technical reasons, in complexity analyses one often works with decision problems rather than optimization problems. A *decision problem* has the binary output-domain $\{yes, no\}$ and is denoted by 'D-' preceding the problem name:

$$\text{D-}\Pi : I_\Pi \rightarrow \{yes, no\}.$$

Given the close relationship between an optimization problem and its associated decision problems (solutions for one can often be transformed to also yield solutions for the other; see Lemmas A and B), focusing on appropriate decision problems does not limit the relevance of the analyses.

The Big-Oh $\mathcal{O}(.)$ notation is used to express an asymptotic upper-bound of a function, i.e., it describes the limiting behavior of a function when the argument tends to large values or infinity. A function $f(x)$ is $\mathcal{O}(g(x))$, if there exist constants $c \geq 0$ and $x_0 \geq 1$ such that $f(x) \leq cg(x)$ for all $x \geq x_0$. Informally $\mathcal{O}(g(x))$ ignores all but the highest order part of the function $f(x)$ (see Table 1 for examples).

To disambiguate between similarly named variables of different problems, we sometimes subscript the variable with the problem's name. E.g., G in Π is denoted as G_Π and G in Θ is denoted as G_Θ.

4.2. Classical computational complexity

Using computational complexity analysis one can study the amount of computational resources required – in most cases

[6] Note that in this formalization it is assumed that the Bayesian network is given as part of the input. It does not model the construction of the Bayesian network based on knowledge and the "assumption of rationality". We hence study the complexity of the inference on the Bayesian network, but without studying the complexity of constructing that network.

Table 1
Examples of Big-Oh.

$f(x)$	$\mathcal{O}(g(x))$
$5x^2$	$\mathcal{O}(x^2)$
$x^5 + x^3 - x + 7342$	$\mathcal{O}(x^5)$
$2^x + x^4 + 16$	$\mathcal{O}(2^x)$
$x! + 2^x + 2x$	$\mathcal{O}(x!)$

time or space – to compute the output of a problem (or model) Π.[7] We are interested in the worst-case scenario (i.e., the maximum time or space required to compute an output), because the computations that models postulate have to be able to be performed using a realistic amount of resources for *all* relevant inputs (see van Rooij, 2008). In worst-case complexity analysis, the computational complexity of a problem or model is expressed as an upper bound on the required resources as a function of the size of the input $\mathcal{O}(g(|i_\Pi|))$, where $i_\Pi \in I_\Pi$. We say a problem Π can be computed in time $\mathcal{O}(g(n))$, where $n = |i_\Pi|$, if there exists an algorithm that computes Π in time $\mathcal{O}(g(n))$.

This section provides the mathematical tools to prove that a problem (or model) has a certain computational complexity, without having to define algorithms that compute it. Any results that follow will thus hold for all implementations of the computational level model, i.e., if a model is intractable then no tractable algorithm exists and if a model is tractable then such an algorithm does exist.

Problems and models can be classified according to their nature and complexity into complexity classes such as P and NP. The class P contains all decision problems—problems that output only *yes* or *no*—that are computable in polynomial time. A problem is computable in polynomial time if there exists an algorithm that computes it in $\mathcal{O}(n^\alpha)$ time for some constant α. The class NP contains all problems for which yes-answers can be verified for their correctness in polynomial time. Trivially $\mathsf{P} \subseteq \mathsf{NP}$ and it is generally believed that $\mathsf{P} \neq \mathsf{NP}$ (Fortnow, 2009). This means that there are problems in NP that are not in P, i.e., problems that cannot be computed in polynomial time by *any* algorithm. These problems thus take super-polynomial (i.e., exponential or worse) amounts of time to compute by any algorithm, a time that is considered impractical for all but small inputs. Hence, the common labels 'intractable' for these problems. To identify which problems fall into this category we use the notion of *hardness*. A problem is hard for a certain complexity class C if it is at least as computationally complex as all hardest problems in C. For example a problem Π is NP-hard if all other problems in NP are at most as hard as Π. Given the assumption that $\mathsf{P} \neq \mathsf{NP}$, proving that a problem is NP-hard also proves that this problem is not in P. In this paper we will assume $\mathsf{P} \neq \mathsf{NP}$.

To prove that a problem is hard for a class we rely on problems for which this is already known and a complexity relation between these problems. This relation is called a *polynomial time (many-one) reduction*. A decision problem Θ is at least as hard as decision problem Π if there exists a polynomial-time reduction from Π to Θ. We say Π reduces to Θ if there exists a function τ that transforms any input i_Π of Π to input $\tau(i_\Pi)$ of Θ such that i_Π is a yes-instance for Π if and only if $\tau(i_\Pi)$ is a yes-instance for Θ. A reduction is a polynomial time reduction if τ is polynomial time computable. We write $\Pi \leq_\tau \Theta$ if Π polynomial time reduces to Θ, i.e., if Θ is at least as hard as Π. Polynomial-time reductions are very powerful and can be used to prove that a problem is NP-hard or in P. If $\Pi \leq_\tau \Theta$ and Π is NP-hard then Θ is NP-hard. Conversely, if $\Pi \leq_\tau \Theta$ and Θ is in P then Π is also in P. For a guide on polynomial time reductions see Box 1 which contains a polynomial-time reduction blueprint for decision problems.

Box 1. Polynomial-time reduction blueprint
The following four steps describe how to define a polynomial-time reduction from decision problem Π to decision problem Θ.

1. Describe a function $\tau(.)$ that transforms any input i_Π for Π into an input i_Θ for Θ, i.e., $i_\Theta = \tau(i_\Pi)$.
2. Assume i_Π is a yes-instance for Π. Show that then also i_Θ is a yes-instance for Θ.
3. Assume i_Θ is a yes-instance for Θ. Show that then also i_Π is a yes-instance for Π.
4. Show that the function τ runs in polynomial-time.

Because the cognitive models we will analyze in this paper are optimization problems, it is useful to know that we can use the blueprint in Box 1 to prove that an optimization problem Π is not polynomial-time computable. To do so, we first define a thresholded decision variant D-Π and prove it is NP-hard. Then by Lemma A (stated below) Π is not computable in polynomial time.

Let $\Pi : I \rightarrow O$ be an optimization problem and D-$\Pi : I \rightarrow \{yes, no\}$ be its corresponding thresholded decision variant, such that D-Π outputs *yes* if $val(o) \geq q$ or $val(o) \leq q$ for some given threshold value q (depending on whether Π is a maximization or minimization problem, respectively) or *no* otherwise. Here $o = \Pi(i)$ and $val(.)$ is assumed to be polynomial-time computable.

Lemma A. *Assuming that* $\mathsf{P} \neq \mathsf{NP}$, *if* D-$\Pi$ *is* NP*-hard then* Π *is not computable in polynomial time.*

Proof. We prove by contradiction. Assume that D-Π is NP-hard and Π is computable in polynomial time. Then, there exists an algorithm A that solves Π in polynomial time. This then also means that there exists an algorithm that solves D-Π in polynomial time. Namely, the algorithm that first calls A to compute $\Pi(i) = o$; then computes $val(o)$ (which by definition can be done in polynomial time); and finally checks if $val(o)$ is greater than or equal to q. Given that D-Π is NP-hard this then implies that P= NP. This contradicts the assumption that $\mathsf{P} \neq \mathsf{NP}$ as stated in the lemma. □

4.3. Parameterized computational complexity

While classical complexity theory provides a methodology to assess the amount of required resources to compute a problem, it fails to explain *what* makes a problem (in-)tractable. This can be done using the theory of parameterized computational complexity developed by Downey and Fellows (1999). Their framework expresses the complexity of a decision problem Π in terms of a set of parameters (or properties) κ of the input. If such a set of parameters κ has an exponential (or worse) contribution to the complexity of the problem, then the problem is tractable provided that the values of the parameters in κ are small enough.

Parameterized complexity theory studies the computational complexity of parameterized problems, i.e., problems with an associated set of parameters denoted κ-Π. As with classical complexity theory, parameterized problems can also be divided into classes. The class FPT contains all parameterized decision problems that are *fixed-parameter tractable* (*fp-tractable*). Formally, this means that there exists at least one algorithm that computes

[7] The type of complexity analyses that we perform in this paper are assumed to be independent of the model of computation used. We refer the interested reader to van Rooij (2008, pp. 945–946 and pp. 963–972) and Aaronson (2005) for explanations and arguments for why this is so. For purposes of this paper, the reader can assume any reasonable model of computation, be it e.g., a classical computer such as a Turing machine or more modern models such as neural networks or probabilistic models of computation.

the output $\Pi(i)$ for a decision problem $\Pi : I \rightarrow \{yes, no\}$ for parameter set κ in $\mathcal{O}(f(\kappa)n^{\alpha})$ time, where f is an arbitrary function (that can be exponential or worse) and α is a constant. Such an algorithm is said to run in *fixed-parameter (fp-) tractable time* for parameter set κ. Observe that if a parameter set κ is found for which Π is fp-tractable then the problem Π can be computed quite efficiently, even for large inputs, provided only that the values of the members of κ are relatively small. In this sense the "unbounded" nature of parameters in κ can be seen as a reason for the (classical) intractability of Π. Therefore we call κ a *source of intractability* of Π (van Rooij, Evans, Müller, Gedge, & Wareham, 2008).

Analogous to $\mathsf{P} \subseteq \mathsf{NP}$, parameterized complexity defines a class W[1] of parameterized decision problems with the property that $\mathsf{FPT} \subseteq \mathsf{W[1]}$. Also analogous to the $\mathsf{P} \neq \mathsf{NP}$ conjecture, it is widely believed that $\mathsf{FPT} \neq \mathsf{W[1]}$ (Downey & Fellows, 1999). Given this conjecture, W[1] contains parameterized decision problems that are *fixed-parameter intractable* (*fp-intractable*). This means that if a parameterized problem κ-Π is W[1]-hard, then there are no algorithms that can compute κ-Π in $\mathcal{O}(f(\kappa)n^{\alpha})$ time. This is useful because it allows us to prove fp-intractability by reduction, similar to the classical approach as we explain next. In this paper we will assume $\mathsf{FPT} \neq \mathsf{W[1]}$.

To prove that a parameterized problem is W[1]-hard, we use *parameterized reductions*. These are similar to polynomial-time reduction, but with two additions. A problem κ-Π parameterized reduces to δ-Θ if: (1) $\Pi \leq \Theta$, (2) all parameters in $d \in \delta$ are a function of one (or more) parameters $k \in \kappa$ and, (3) the parameterized reduction runs in fp-tractable time for κ. For a guide on parameterized reductions see Box 2 which contains a parameterized reduction blueprint.

Box 2. Parameterized reduction blueprint
The following five steps describe how to define a parameterized reduction from decision problem κ-Π to decision problem δ-Θ.

1. Describe a function $\tau(.)$ that transforms any input i_{Π} for Π into an input i_{Θ} for Θ, i.e., $i_{\Theta} = \tau(i_{\Pi})$.
2. Assume i_{Π} is a yes-instance for Π. Show that then also i_{Θ} is a yes-instance for Θ.
3. Assume i_{Θ} is a yes-instance for Θ. Show that then also i_{Π} is a yes-instance for Π.
4. Show that the function τ runs in fp-tractable time relative to κ.
5. Show that each parameter $d \in \delta$ can be expressed by a function $f(\kappa)$.

Similar to classical reductions, parameterized reductions between appropriate parameterized decision problems can be used to show that an optimization problem is not fp-tractable. Let κ-$\Pi : I \rightarrow O$ be an optimization problem with parameter κ and κ-D-$\Pi : I \rightarrow \{yes, no\}$ be its corresponding thresholded parameterized decision variant, where κ-D-Π outputs *yes* if $val(o) \geq q$ and $f(o) \leq k$ for some given threshold value q (depending on whether κ-Π is a maximization or minimization problem, respectively) or *no* otherwise. Here $o = \kappa$-$\Pi(i)$ and $val(.)$ is assumed to be polynomial-time computable.

Lemma B. *Assuming that* $\mathsf{FPT} \neq \mathsf{W[1]}$, *if* κ-D-Π *is* W[1]*-hard then* κ-Π *is not fp-tractable.*

Proof. We prove by contradiction. Assume that κ-D-Π is W[1]-hard and κ-Π is computable in fp-tractable time. Then, there exists an algorithm A that solves κ-Π in fp-tractable time. This then also means that there exists an algorithm that solves κ-D-Π in fp-tractable time. Namely, the algorithm that first calls A to compute $\Pi(i) = o$; then computes $val(o)$ (which by definition can be done in polynomial time); and finally checks if $val(o)$ is greater than or equal to q. Given that κ-D-Π is W[1]-hard this then implies that $\mathsf{FPT} = \mathsf{W[1]}$. This contradicts the assumption that $\mathsf{FPT} \neq \mathsf{W[1]}$ as stated in the lemma. □

5. Computational complexity analysis

Having introduced computational complexity theory and the computational-level model of Bayesian inverse planning we are now in a position to analyze the computational resource requirements (i.e., computational complexity) of the BIP model. Based on the results of this analysis we will be able to draw conclusions on whether or not the BIP model can explain when and why people can perform goal inferences quickly.

5.1. Classical computational complexity analysis

As mentioned in Section 4 it is often difficult to directly prove that an optimization problem (such as BIP) is computationally intractable. In this subsection we will use an easier strategy by proving that the decision version of BIP is NP-hard. Then by Lemma A we can prove that the original (i.e., the optimization version) of BIP is not computable in polynomial time.

We first have to define this decision version of BIP which we call DECISION-BIP (or D-BIP for short). The output of D-BIP is *yes* or *no* based on the marginal probability of $\mathbf{g}$, $\mathbf{s}$ and $\mathbf{a}$. If there exists a $\mathbf{g}$ such that its marginal probability (i.e., $\Pr(\mathbf{g}, \mathbf{s}, \mathbf{a})$) is larger than a certain threshold variable q then D-BIP will output *yes* and it will output *no* otherwise.

Using the marginal probability makes the definition of the decision model different from the original definition of BIP which is based on conditional probability $\Pr(\mathbf{g} \mid \mathbf{s}, \mathbf{a})$. However, we note that this design choice does not change the model because maximizing the conditional probability over $\mathbf{g}$ is the equivalent of maximizing the marginal probability over $\mathbf{g}$.[8]

DECISION-BIP (D-BIP)
Input: Same input as BIP plus an integer $0 \leq q < 1$.
Question: Does there exist a joint value assignment $\mathbf{g}$ to all goal variables in $\mathbf{G}$ given $\mathbf{s} \cup \mathbf{a}$, such that $\Pr(\mathbf{g}, \mathbf{s}, \mathbf{a}) > q$?

The next step is to prove that D-BIP is NP-hard. We will construct a polynomial time reduction from D-CLIQUE—a known NP-hard problem (Garey & Johnson, 1979)—to D-BIP according to the blueprint in Box 1. This reduction will prove that D-BIP is NP-hard, because D-CLIQUE is NP-hard.

DECISION-CLIQUE (D-CLIQUE)
Input: An undirected graph $G = (V, E)$ and an integer $k > 0$.
Question: Does there exist a *clique* in G of size k? Here, a clique is a subset $V' \subseteq V$ such that $\forall_{u,v \in V'}[(u, v) \in E]$ and the size of a clique is the size of the subset V'.

Theorem A. D-BIP *is* NP*-hard.*

Proof. The proof presented here is a conceptual sketch, albeit a detailed one, to provide the reader with an overview of the principles that are involved. For an even more detailed proof we refer to Appendix A.1.

Step 1. Given an instance $\langle \mathcal{G} = (V, E), k \rangle$ of D-CLIQUE, transform it to an instance $\langle \mathcal{B}, \mathbf{a}, \mathbf{s}, q \rangle$ of D-BIP. This resulting D-BIP-instance can be seen as a 'machine' that solves D-CLIQUE. The transformation can be used to transform any D-CLIQUE-instance. To illustrate this transformation we use the D-CLIQUE-instance in Fig. 4, assuming

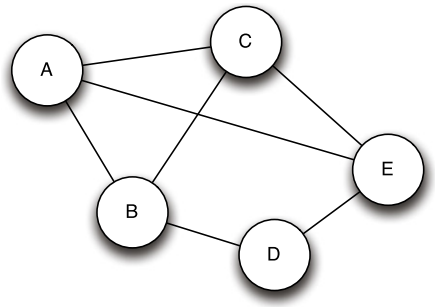

Fig. 4. An example CLIQUE-instance with five vertices $V = \{A, \ldots, E\}$ and their edges $E = \{(A, B), (A, C), (B, C), (B, D), (C, E), (D, E)\}$.

that $k = 3$, as an example. We will transform it to a D-BIP-instance depicted in Fig. 5.

We start by creating a Bayesian network in accordance with the BIP definition by creating the following variables:

- k goal variables: these variables encode possible subsets of vertices and their domain consists of all the vertices in the D-CLIQUE-instance;
- $(k-1) + k(k-1)/2$ action variables: these Boolean variables will be used to check if a subset encoded in the goal variables satisfies the requirements of a clique of size k;
- $1 + (k-1) + k(k-1)/2$ state variables: these Boolean variables will be used to conjoin all the action variables such that a joint value assignment is only possible if all requirements of a clique are met.

The values of the goal variables must satisfy the following two requirements to allow these variables to encode possible cliques. First, no two variables can encode the same vertex. Second, the vertices encoded by each possible pair in the subset should be connected by an edge. To check for the first requirement we define an arbitrary ordering on the vertices in V. Based on this ordering we define $(<)$-nodes. There are $k-1$ actions variables that act as $(<)$-nodes and each of these variables depends on, in order, two goal variables. The probability that $(<)$-nodes are *true* is 1 if and only if the vertex encoded in the first goal variable is ordered lower than the one encoded in the second goal variable. Otherwise these nodes have a probability of 1 being *false*. This ensures that only value assignments to the goal variables that encode different vertices in each goal have a probability of 1, all other value assignments have a probability of 0.

To check for the second requirement the remaining $k(k-1)/2$ action variables are defined as $(\in E)$-nodes. Each of these action variables is connected to a unique pair of goal variables. The probability that $(\in E)$-nodes are *true* is 1 if and only if the vertex pair encoded in the goal variables has an edge between them in G. Otherwise these nodes have a probability of 1 being *false*. This ensures that any value assignment to the goal variables encodes a clique with probability 1, and all other (non-clique) value assignments have a probability of 0.

The domain of the first state variable S_1 is set to a single value *true* (i.e., we make it a dummy variable) and dependencies between action and goal variables not mentioned above do not exist in the constructed instance. Finally, to complete the instance of D-BIP, all variables in $\mathbf{S}$ and $\mathbf{A}$ are observed to be *true* and $q = 0$.

To prove that the transformation above is a reduction, we must show that the answer to the given instance of D-CLIQUE is *yes* if and only if the answer to the constructed instance of D-BIP is *yes*. This corresponds to steps 2 and 3 below.

Step 2. If the output for the given instance of D-CLIQUE is *yes*, then there exists at least one subset $V' \subseteq V$ such that $|V'| = k$ and for all $u, v \in V'$, $(u, v) \in E$. The fact that V' is a clique means that all elements in V' are unique and there is an edge between each pair of distinct vertices in V'. Note that only joint value assignments with these properties have probability $\Pr(\mathbf{g}, \mathbf{s}, \mathbf{a}) > 0$ in the constructed instance of D-BIP, due to the $(<)$-nodes created in Step 1 which enforce that the variables in $\mathbf{g}$ encode k distinct variables, and the $(\in E)$-nodes created in Step 1 that enforce which all these variables are connected. Every other joint value assignment has probability $\Pr(\mathbf{g}, \mathbf{s}, \mathbf{a}) = 0$. Thus, if there is a k-clique in G, there exists at least one joint value assignment $\mathbf{g}$ with probability $p > 0$, namely the joint value assignments that encode these cliques in $\mathbf{g}$. Given the structure of $\mathcal{B}$, this implies that $\Pr(\mathbf{g}, \mathbf{s}, \mathbf{a}) > 0$ and thus that the output of the constructed instance of D-BIP is *yes*.

Step 3. If the output for the constructed instance of D-BIP is *yes*, then there is an assignment $\mathbf{g}$ to $\mathbf{G}$ such that $\Pr(\mathbf{g}, \mathbf{s}, \mathbf{a}) > 0$. Given the structure of $\mathcal{B}$ and given that all action and state variables are observed to be *true*, the only possible joint value assignment $\mathbf{g}$ is the one in which the values of the goal variables are not only distinct (ensured by the $(<)$-nodes), but also where these values correspond to a set of vertices such that every distinct pair of vertices encoded in $\mathbf{g}$ is connected by an edge in $\mathcal{G}$ (ensured by the $(\in E)$-nodes). Thus, the set of vertices corresponding to the goal variable values in $\mathbf{g}$ corresponds to a k-clique in $\mathcal{G}$, which means that the output of the given instance of D-CLIQUE is *yes*.

Step 4. As the number of conditional probability tables that need to be constructed above is proportional to the total number of variables (which is $|\mathbf{S}| + |\mathbf{A}| + |\mathbf{G}| = (1 + (k-1) + k(k-1)/2) + ((k-1) + k(k-1)/2) + k$) and each table involves at most 3 variables with at most $|V|$ values per variable (giving tables with at most $|V|^3$ entries), this construction can be done in time polynomial in the size of the given instance of D-CLIQUE. □

From Theorem A, Lemma A and the polynomial time computability of $\Pr(\mathbf{g}, \mathbf{s}, \mathbf{a})$ (van der Gaag, 1990) we can now conclude that BIP is not computable in polynomial time.

Result 1. BIP *is not computable in polynomial time (unless* P = NP*).*

5.2. Parameterized complexity analysis of BIP

In the previous section we showed that BIP is computationally intractable. We next study the parameterized complexity of BIP for different sets of parameters—using the methodology from Section 4.3—in an attempt to identify domain-specific constraints that render this (otherwise intractable) model computational tractable. The parameters we consider are the number of actions, the number of goals, the probability of the most likely joint value assignment and the maximum cardinality of the goal variables (see Table 2).

Table 2
An overview of all the parameters we consider in this paper, their description and corresponding value in the example from Fig. 2.

Parameter	Description	Value
$\lvert\mathbf{A}\rvert$	The number of actions, informally the length of the observed behavior	3
$\lvert\mathbf{G}\rvert$	The number of goals	2
$1-p$	One minus the probability of the most likely joint value assignment to $\mathbf{G}$, informally the higher $1-p$ the more ambiguous the interpretations of the observations are.	N/a
g	The maximum cardinality of any goal variable	3

Table 3 presents an overview of the parameterized computational complexity of BIP with respect to (combinations of) the parameters in Table 2. The remainder of this section details the proofs

[8] Maximizing $\Pr(\mathbf{g} \mid \mathbf{s}, \mathbf{a})$ is equivalent to maximizing $\Pr(\mathbf{g}, \mathbf{s}, \mathbf{a})$ because $\Pr(\mathbf{g} \mid \mathbf{s}, \mathbf{a}) = \Pr(\mathbf{g}, \mathbf{s}, \mathbf{a}) / \Pr(\mathbf{s}, \mathbf{a})$ and $\Pr(\mathbf{s}, \mathbf{a})$ is constant.

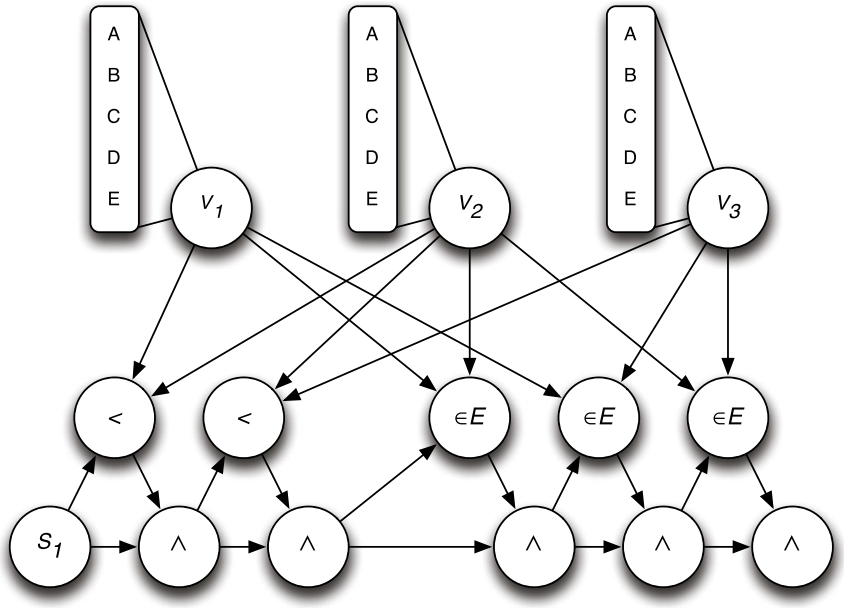

Fig. 5. An example transformation of the CLIQUE-instance in Fig. 4 to a BIP-instance. This particular instance is structured such that it is a yes-instance if and only if there exists a subset of $\{A, \ldots, E\}$ of size three that is a clique. The $(<)$-nodes ensure that only subsets of V are viable value assignments (viz., $\{(A, B, C), (A, B, D), (A, B, E), (A, C, D), (A, C, E), (A, D, E), (B, C, D), (B, C, E)\}$). The $(\in E)$-nodes ensure that only subsets that are cliques are viable value assignments (viz., $\{(A, B, C), (A, C, E)\}$). Furthermore, the rounded boxes denote the domains of variables.

Table 3
An overview of the parameterized complexity of BIP. The subscript numbers refer to the respective results. Question marks label parameter sets for which the computational complexity remains an open question (see Open question 1).

Parameter	–	g	$\|\mathbf{G}\|$	$\|\mathbf{G}\|, g$
–	NP-hard$_1$	fp-int.	fp-int.	fp-trac.$_4$
$\|\mathbf{A}\|$	fp-int.	fp-int.$_3$	fp-int.$_2$	fp-trac.
$1-p$	?	fp-trac.$_5$	?	fp-trac.
$\|\mathbf{A}\|, 1-p$	?	fp-trac.	?	fp-trac.

of some of these results. The cells in Table 3 without explicit proofs follow from the results with explicit proofs by Observation 1.

Observation 1. If Π is a problem that is fp-intractable for the parameter set κ, then Π is also fp-intractable for any subset $\kappa' \subseteq \kappa$. Conversely, if Π is a problem that is fp-tractable for the parameter set κ, then Π is also fp-tractable for any super set $\kappa' \supseteq \kappa$.

The first set of parameters we investigate is $\{|\mathbf{A}|, |\mathbf{G}|\}$. We will show, via parameterized reduction from $\{k\}$-D-CLIQUE—a known W[1]-hard problem (Downey & Fellows, 1999)—to $\{|\mathbf{A}|, |\mathbf{G}|\}$-D-BIP and Lemma B, that the BIP model is fp-intractable.

DECISION CLIQUE (D-CLIQUE)
Input: A undirected graph $G = (V, E)$ where V is ordered and $k \in \mathbb{N} > 0$.
Parameters: k
Question: Does there exist a *clique* of size k? Here, a clique is a subset $V' \subseteq V$ such that $\forall_{u,v \in V'}[(u, v) \in E]$ and the size of a clique is the size of the subset V'.

Corollary A. $\{|\mathbf{A}|, |\mathbf{G}|\}$-D-BIP *is* W[1]*-hard.*

Proof. This proof follows the parameterized reduction blueprint in Box 2.

Step 1. Given an instance $\langle \mathcal{G} = (V, E), k\rangle$ of $\{k\}$-D-CLIQUE, translate it to an instance $\langle \mathcal{B}, \mathbf{a}, \mathbf{s}, q\rangle$ of $\{|\mathbf{A}|, |\mathbf{G}|\}$-D-BIP exactly the same as the transformation in Step 1 in the proof of Theorem A.

Steps 2 & 3. These steps are the same as Steps 2 and 3 in the proof of Theorem A, because the instance transformation is the same.

Step 4. In Step 4 in the proof of Theorem A we proved the transformation runs in polynomial time. This means that the transformation also runs in fp-tractable time for parameter set $\{|\mathbf{A}|, |\mathbf{G}|\}$, when we ignore the parameter set.

Step 5. The transformation in Step 1 ensures that $|\mathbf{A}| = (k-1) + k(k-1)/2$ and $|\mathbf{G}| = k$. □

From Corollary A, Lemma B and the polynomial-time computability of the marginal probability $\Pr(\mathbf{g}, \mathbf{s}, \mathbf{a})$ we can now conclude the computational complexity of BIP with respect to the parameters in this set.

Result 2. BIP *is fp-intractable for the parameter set* $\{|\mathbf{A}|, |\mathbf{G}|\}$ *(unless* FPT = W[1]*).*

The second parameter set we investigate is $\{|\mathbf{A}|, g\}$. We will show that $\{|\mathbf{A}|, g\}$-BIP is also fp-intractable. The proof consists of a reduction from $\{k\}$-D-CLIQUE to $\{|\mathbf{A}|, g\}$-D-BIP.

Corollary B. $\{|\mathbf{A}|, g\}$-D-BIP *is* W[1]*-hard.*

Proof. The proof presented here is a conceptual sketch, albeit a detailed one, to provide the reader with an overview of the principles that are involved. For an even more detailed proof we refer to Appendix A.2.

Step 1. Given an instance $\langle \mathcal{G} = (V, E), k\rangle$ of k-D-CLIQUE, translate it to an instance $\langle \mathcal{B}, \mathbf{a}, \mathbf{s}, q\rangle$ of $\{|\mathbf{A}|, g\}$-D-BIP similar to the transformation in Step 1 in the proof of Theorem A, but change the following (see also Fig. 6). Instead of k goal variables, create k blocks of $\lceil \log_2 |V| \rceil$ ordered Boolean goal variables. We use these blocks of Boolean goal variables to encode, in binary, the vertices of the CLIQUE instance (e.g., in our example 000 encodes A, 001 encodes B, etc.). Whereas previously (in the proof of Theorem A) action variables depended on single goal variables that could encode vertices from the clique instance, now action variables are dependent on all goal variables in a block that together can encode clique vertices.

Steps 2 & 3. These steps are the same as Steps 2 and 3 in the proof of Theorem A, modulo the replacement of multi-valued goal variables by blocks of Boolean goal variables.

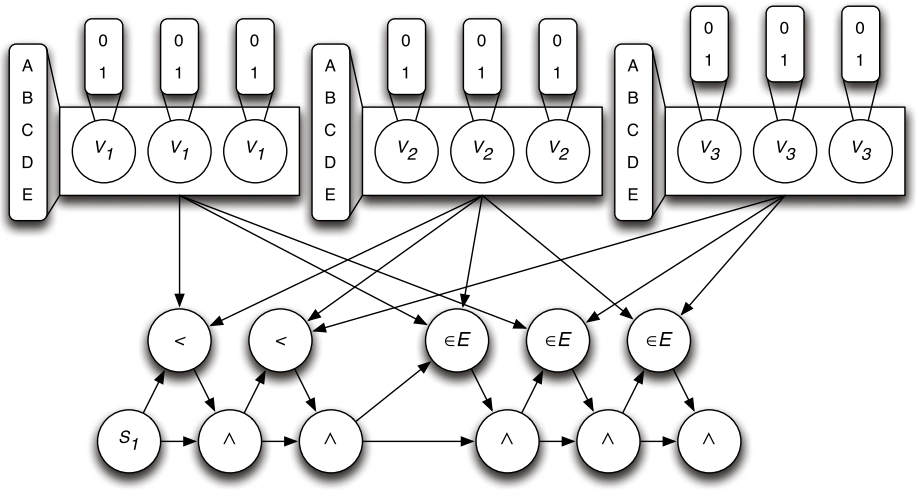

Fig. 6. An example parameterized reduction from the CLIQUE instance in Fig. 4 to BIP. We note that in this figure the dependency of an action variable on a 'block of goal variables' means that it is dependent on all goal variables in that block. Furthermore, the rounded boxes attached to variables denote their domains and the rounded boxes attached to blocks denote the possible clique vertices that can be encoded in that block.

Step 4. The transformation in Step 1 runs in fp-tractable time for the parameter set $\{|\mathbf{A}|, g\}$, as the number of variables is a function of parameter k of D-CLIQUE, i.e., $|\mathbf{S}| + |\mathbf{A}| + |\mathbf{G_C}| = (1 + (k-1) + k(k-1)/2) + ((k-1) + k(k-1)/2) + k\lceil \log_2 |V| \rceil$, and each conditional probability table involves at most $3\lceil \log_2 |V| \rceil$ Boolean variables (and hence has at most $2^{3\lceil \log_2 |V| \rceil} \leq 8|V|^3$ table entries).

Step 5. The transformation ensures that $|\mathbf{A}| = 1 + (k-1) + k(k-1)/2 + 1$ and $g = 2$. □

From Corollary B and Lemma B we can now conclude:

Result 3. BIP *is fp-intractable for the parameter set* $\{|\mathbf{A}|, g\}$ *(unless* FPT = W[1]*).*

We note that the computational complexity for the parameter set $\{|\mathbf{G}|, |\mathbf{A}|, 1-p\}$ remains an open question.

Open question 1. Currently there is no known proof that $\{|\mathbf{G}|, |\mathbf{A}|, 1-p\}$-BIP is fp-intractable and it seems that none of our proofs can be trivially extended to construct such a proof. We note, however, that the fastest known algorithm solving BIP is not fp-tractable with respect to the parameter set $\{|\mathbf{G}|, |\mathbf{A}|, 1-p\}$ (Bodlaender, van den Eijkhof, & van der Gaag, 2002). This suggests that an fp-tractable algorithm for $\{|\mathbf{G}|, |\mathbf{A}|, 1-p\}$-BIP is, at best, non-trivial to construct and, at worst, impossible to construct. In light of these considerations, we conjecture that $\{|\mathbf{G}|, |\mathbf{A}|, 1-p\}$-BIP is fp-intractable until proven otherwise. If the conjecture were to be shown correct, then by Observation 1 it would follow that BIP is also fp-intractable for parameter sets $\{1-p\}$, $\{|\mathbf{A}|, 1-p\}$ and $\{1-p, |\mathbf{G}|\}$.

We have also derived fp-tractability results for the parameter sets $\{|\mathbf{G}|, g\}$ and $\{1-p, g\}$. In order to show these results we will use the notion of treewidth, defined below. We will show that an fp-tractable algorithm exists via parameterized reduction to MPE, for which known fp-tractable algorithms exist that require exponential time in the treewidth of a BN and the maximum domain size of a variable.

Definition 1. A *tree-decomposition* (Robertson & Seymour, 1986) of a graph $G = (V, E)$ is a pair $\langle T, \mathcal{X} \rangle$, where $T = (I, F)$ is a tree and $\mathcal{X}$ is a set of subsets of V, one for each node of T, such that:

1. $\mathcal{X}$ is a cover of V, $\cup\mathcal{X} = V$;
2. each edge in E is contained in a set in $\mathcal{X}$, $\forall_{(x,y)\in E}\exists_{X\in\mathcal{X}}[x \in X \wedge y \in X]$;
3. and each bag on a path F^+ between two bags contains the disjunction of those two bags, $\forall_{(X,Y)\in F^+ \wedge (Y,Z)\in F^+}[X \cup Z \subseteq Y]$.

Definition 2. The *treewidth* (tw) (Robertson & Seymour, 1986) of a BN $\mathcal{B}$ is defined as the minimum width over all tree-decompositions of the moralized graph[9] of $\mathcal{B}$, where the width of a tree-decomposition $(\mathcal{X}, F)$ is equal to the size of a largest bag in $\mathcal{X}$ minus 1, $tw(\mathcal{B}) = \max_{X\in\mathcal{X}} |X| - 1$.

MOST PROBABLE EXPLANATION (MPE)
Input: A probabilistic network $\mathcal{B} = (\mathbf{G}, \Gamma)$, where $\mathbf{G} = (\mathbf{V}, \mathbf{A})$, $\mathbf{V}$ is partitioned into a set of evidence nodes $\mathbf{E}$ with a joint value assignment $\mathbf{e}$ and an explanation set $\mathbf{M}$ (i.e., $\mathbf{E} \cup \mathbf{M} = V$ and $\mathbf{E} \cap \mathbf{M} = \varnothing$).
Parameters: treewidth tw (see Definition 2); max. domain size of variables d
Output: The most probable joint value assignment $\mathbf{m}$ to the nodes in $\mathbf{M}$ given evidence $\mathbf{e}$.

Theorem B. $\{|\mathbf{G}|, g\}$-BIP *is fp-tractable.*

Proof. To prove $\{|\mathbf{G}|, g\}$-BIP is fp-tractable it suffices to provide a parameterized reduction to a fp-tractable problem, in this case $\{tw, d\}$-MPE (Kwisthout, 2011). Again, we use the blueprint from Box 2.

Step 1. Given an instance $\langle \mathcal{B}, \mathbf{a}, \mathbf{s} \rangle$ of $\{|\mathbf{G}|, g\}$-BIP transform it to an instance $\langle \mathcal{B}, \mathbf{e} \rangle$ of $\{tw, d\}$-MPE as follows:

i. Copy the Bayesian network $\mathcal{B}_{BIP}$ of the instance of $\{|\mathbf{G}|, g\}$-BIP to $\mathcal{B}_{MPE}$ of the instance of $\{tw\}$-MPE.
ii. $\mathbf{G}$ is partitioned into $\mathbf{E}$ and $\mathbf{M}$, where $\mathbf{E} = \mathbf{A} \cup \mathbf{S}$ and $\mathbf{M} = \mathbf{G}$.
iii. The evidence $\mathbf{e} = \mathbf{a} \cup \mathbf{s}$.

Step 2. If $\langle \mathcal{B}, \mathbf{a}, \mathbf{s} \rangle$ is a yes-instance, then there exists a joint value assignment $\mathbf{g}$ to the goal variables $\mathbf{G}$ in the BIP instance that

[9] A moralized BN is the undirected graph that is formed by first adding arcs (of arbitrary direction) between all pairs of parents in the directed graph and then making all edges undirected. We use the phrase 'moralized graph' to refer to the moralization of the graph of a BN.

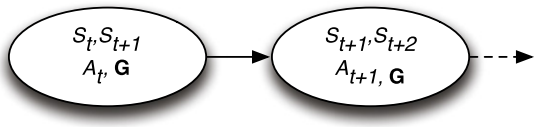

Fig. 7. The tree decomposition of BIP networks (assuming complete dependency of all actions on all goals, i.e., the maximum the definition allows). Each bag contains two sequential states, an action and all goals. This means that the size of each bag is $3 + n$, where n is the number of goals. The treewidth then is $2 + n$, viz., the size of the largest bag minus one.

has the highest posterior probability. Then, because $\mathbf{A} \cup \mathbf{S} = \mathbf{E}$ and $\mathbf{G} = \mathbf{M}$ and both Bayesian networks are equal, the joint value assignment $\mathbf{m}$ with the highest posterior probability for $\mathbf{M}$ is equal to $\mathbf{g}$.

Step 3. Conversely to Step 2, if $\langle \mathcal{B}, \mathbf{e} \rangle$ is a yes-instance, then there exists a joint value assignment $\mathbf{m}$ to $\mathbf{M}$ that has the highest posterior probability. Then, because $\mathbf{E} = \mathbf{A} \cup \mathbf{S}$ and $\mathbf{M} = \mathbf{G}$ and both Bayesian networks are equal, the joint value assignment $\mathbf{g}$ with the highest posterior probability for $\mathbf{G}$ is equal to $\mathbf{m}$.

Step 4. The transformation runs in time linear to the size of the input, and thus it is fp-tractable.

Step 5. The transformation ensures that $d = g$, the maximum domain size in the MPE-instance is equal to the maximum domain size of the goal variables in the BIP-instance. The transformation also ensures that the treewidth of the MPE-instance tw is a function of the number of goal variables in the BIP-instance $|\mathbf{G}|$:

i. For a BIP-instance with 1 goal variable, a tree-decomposition such as the one in Fig. 7 can be constructed, meaning such an instance has a treewidth of 3.
ii. For a BIP-instance with $n + 1$ goal variables, a similar tree-decomposition can be constructed. However, each bag will at least also contain the extra $(n+1)$th goal variable, because all actions can depend on that goal variable.

From these inductive steps we can conclude that the tree-width increases as the number of goals increase, i.e., treewidth is a (linear) function of the number of goals. □

Result 4. $\{|\mathbf{G}|, g\}$-BIP *is fp-tractable.*

The final parameter set we consider is $\{1 - p, g\}$. We show that an algorithm for BIP exists that runs in time non-polynomial only in $1 - p$ and g.

Theorem C. $\{1-p, g\}$-BIP *can be solved in time non-polynomial only in* $1 - p$ *and* g.

Proof. An algorithm exists that solves the decision version of MPE for Boolean variables in $\mathcal{O}(2^{(\log_2 p/\log_2 1-p)} \cdot n)$ (see Bodlaender et al., 2002; Kwisthout, 2011). The algorithm proposed by Bodlaender et al. can easily be extended to work for variables that have larger domain sizes and to return the most probable value assignment.

The extended algorithm takes for its input a probabilistic network B, the set of observed O variables along with their assignment a_O, an array A, a probability p, and an integer i. The array $A[1 \ldots n]$ is used to store values for the variables in the network; the element $A[i]$ stores a value for the variable V_i. We assume, without loss of generality, that in the first call to the procedure the elements of the array A are initialized to the first value of the variable $A[i] = v_i^0$. The input parameter i denotes the level in the search tree that is currently being investigated; at level i, the search process has fixed the values for the variables $V_1 \ldots V_i$. In the first call to the procedure, this parameter is initialized to 0. The following pseudo-code summarizes the extended algorithm.

```
procedure MPE(B; O; a_O; A; p; i)
  if p > 1 then return ∅ endif;
  if i = n then return A endif;
  for c = 1 to |Ω(V_{i+1})|
    set q_c = Pr(V_{i+1} = v^c_{i+1} | V_1 = A[1], ..., V_i = A[i]);
    if (V_{i+1} ∉ O or (V_{i+1} ∈ O and V_{i+1} = v^c_{i+1})) and
    q_c ≠ 0 and q_c ≥ p then
      A[i + 1] := v^c_{i+1};
      if MPE(B; O; a_O; A; p = q_c; i + 1) ≠ ∅ then
        return A;
      endif;
    endif;
  end for;
return ∅;
```

This algorithm runs in $\mathcal{O}(g^{(\log_2 p/\log_2 1-p)} \cdot n) = \mathcal{O}(f(g, p) \cdot n)$, which follows from the runtime of the original algorithm. This means that MPE can be solved in time non-polynomial only in $1-p$ and g, i.e., efficiently when both $1 - p$ and g are small.

The transformation in the proof of Theorem B also proves that BIP is a special case of MPE. Because this transformation leaves the probability of the most likely explanation $1 - p$ and the maximum size of the domain of variables intact, the transformation runs in polynomial time, and because MPE can be solved in time non-polynomial only in $1 - p$ and g (using the algorithm above), we can conclude that BIP can also be solved in time non-polynomial only in $1 - p$ and g. □

Result 5. $\{1 - p, g\}$-BIP *is fp-tractable.*

6. Discussion

The ability to recognize which goals motivate the behavior of people we observe seems, prima facie, to be explained by an appeal to models of abduction, i.e., by models which postulate that people infer those goals that 'best explain' the observed behavior. That being said, models of abduction—such as the Bayesian model in this paper—are in general known to be computationally intractable (NP-hard, see Abdelbar & Hedetniemi, 1998; Bylander et al., 1991; Shimony, 1994), and it is thus not possible that people can quickly perform goal inferences as posited by such models. The intractability of (Bayesian) models of abduction has been used to raise doubts about the computational feasibility of such models, even leading some researchers to reject such models altogether (e.g., Gigerenzer et al., 2008). Rather than rejecting abduction as an account of goal inference, we posit that the speed of goal inference *can* be explained by models of abduction by appealing to the right set of situational constraints. Namely, we view the intractability of models as the result of an overgeneralization: i.e., intractable models—when unconstrained—include all logically possible situations, even ones that do not correspond to real world situations in which humans are able to quickly infer goals. It is well known that a computation that is intractable for an unconstrained input domain can be tractable for a constrained input domain. Hence, by identifying the ways in which a model overgeneralizes its input domain, and removing the overgeneralization by incorporating the real-world constraints into the model, one can explain when and why speedy goal inference is possible.

At the beginning of this paper we set out two main objectives: (1) Show how cognitive modelers can use complexity-theoretic methodology to identify situational constraints that render a computational level model tractable and (2) show how this methodology can help explain when and why speedy goal inferences are possible in the real world. In this paper we used the Bayesian inverse planning (BIP) model as a case study (Baker

Table 4
An illustration of how the time needed to compute output for (resp.) polynomial, exponential and fixed-parameter tractable models scales qualitatively differently for different input sizes n. We assume 10,000 computational steps per second for illustrative purposes. This assumption has a limited effect on the actual time required to compute output for intractable models (cf. van Rooij, 2008). For instance, even when assuming 10^{15} computational steps per second an exponential (2^n) time computation for an input size of 500 would still take $2.5 \cdot 10^{129}$ years. The table clearly illustrates that the time required for fixed-parameter tractable models is orders of magnitude less for small parameters than for intractable (e.g., exponential) models. Hence, fixed-parameter tractability results explain when and why speedy computation is possible.

n	Tractable	Intractable	Fixed-parameter tractable			
	$\mathcal{O}(n^2)$	$\mathcal{O}(2^n)$	$\mathcal{O}(g^{\lvert G\rvert}n^2)$		$\mathcal{O}(g^{(\log_2 p/\log_2 1-p)}n)$	
			$g=2, \lvert G\rvert=2$	$g=2, \lvert G\rvert=5$	$g=2, p=0.8$	$g=2, p=0.6$
5	2.5 ms	3.2 ms	10 ms	80 ms	0 s	0 s
20	40 ms	1.5 min	160 ms	1.28 s	40 ms	70 ms
50	250 ms	85 684 yr	1 s	8 s	280 ms	370 ms
100	1 s	$9.6 \cdot 10^{19}$ yr	4 s	32 s	1.1 s	1.5 s
250	6 s	$1.4 \cdot 10^{65}$ yr	25 s	3 min	6.9 s	9.2 s
500	25 s	$2.5 \cdot 10^{140}$ yr	100 s	13 min	28 s	37 s

et al., 2009, 2007). In the remainder of this section, we discuss how our complexity analyses have identified constraints that do and do not render BIP tractable, how empirical predictions can be derived from these complexity results, and the extent to which the results may or may not apply to other variants of the BIP model.

6.1. Implications

To investigate when and why speedy goal inferences are possible in the real world, we analyzed the computational complexity of the Bayesian inverse planning (BIP) model. We gave a formal definition of the BIP model at Marr's computational level. This definition generalizes the simplest model proposed by Baker et al. (2009, 2007) (what those authors refer to as M1), in the sense that it can accommodate people's ability to infer multiple goals simultaneously.

Analysis of the BIP model revealed that it is intractable (NP-hard; Result 1). This means that the model by itself is computationally unfeasible and thus it cannot explain why humans are able to quickly infer goals in the real world. So if the BIP model is to account for human goal inference at all it must be that in those situations where humans are able to quickly and effortlessly infer multiple simultaneous goals, specific constraints apply that render the BIP model tractable.

To find these specific constraints we used a methodology for identifying sources of intractability in NP-hard computational models (see van Rooij, Evans et al., 2008) and we derived several more theoretical results. For instance, we ruled out the possibility of explaining speedy real world goal inferences solely by an appeal to a small number of values per goal node g, modeling situations where each of the observed person's goals has only a few possible values (Result 3). Similarly, we ruled out that the speed of such inferences could be explained by an appeal to small values of $|\mathbf{A}|$, modeling situations where goals can be inferred using only few observations (Result 3). Even appealing to both constraints at the same time cannot factor in such an explanation (also Result 3). Furthermore, an appeal to a small number of goal nodes $|\mathbf{G}|$—modeling situations where only few goals have to be inferred—also cannot explain the speed (Result 2), not even when combined with constraints on $|\mathbf{A}|$.

However, besides these negative theoretical results, we also have derived two important positive results. For one, we established that as long as *both* the number of goals that are simultaneously pursued $|\mathbf{G}|$ is not too large and there are only a small number of values per goal node g, then goal inference is tractable under the BIP model (Result 4). Secondly we have shown that goal inference is tractable under the BIP model whenever the most likely (combination) of goals has a much higher probability given the observations than all alternatives, i.e., p is not too far from 1, and there are only a small number of values per goal node g (Result 5). The analysis reveals that as long as either (or both) of these situations are in effect, other properties of the input—e.g., the length of the observed behavior $|\mathbf{A}|$—have little impact on the time complexity of goal inference.

Whereas our negative theoretical results are useful to clarify that tractability is not a property that is trivially achieved when trying to characterize the cognitive capacity for goal inference, our positive results show that a model of goal inference can nevertheless be rational, Bayesian, and tractable. Note that our computational-level analyses directly suggest a way of explaining the speed of goal inference. This is illustrated in Table 4. This table shows that goal inference under the BIP model can be performed fast when the right set of situational constraints apply (in this case, either small g and $|\mathbf{G}|$ or small g and $1-p$). In fact, under those situational constraints the inference can be performed orders of magnitude faster than without those constraints, bringing the expected speed of the inference within a qualitatively plausible range (i.e., order of seconds, rather than centuries). Notably, our explanation of speed of goal inference is not by an appeal to specific algorithms (for which it is often difficult to determine that the human brain exactly implements them), but rather by an appeal to situational constraints that human minds/brains can in principle tractably exploit.

6.2. Predictions

Our theoretical results not only provide a way to explain the speed of goal inferences in the real world, but they also lead to novel predictions that can be tested in the lab. The type of predictions derived from complexity-theoretic analyses significantly differ from more common types of predictions. For example, they do not predict the goals people infer for a given situation, but they predict people's performance given certain properties of the situation. Our analyses reveal that the BIP model predicts that humans can in principle quickly infer goals when either (or both) of the following sets of situational constraints are in effect:

i. The number of goals that are simultaneously pursued $|\mathbf{G}|$ are small *and* there are only a small maximum number of values per goal variable g;
ii. The most likely (combination) of goals has a much higher probability given the observations than all alternatives (i.e., $1-p$ is small) *and* there are only a small maximum number of values per goal variable g.

This in turn leads to the prediction that if people perform goal inference in situations different from (i) and (ii), they cannot exploit these constraints to tractably infer goals. As a result their performance is predicted to break down (at least in speed, accuracy or both) as the otherwise constrained situational properties increase in value. If the prediction were to be confirmed then this

would provide corroborative support for the BIP model of goal inference, and validate that our theoretical results help explain the tractability of human goal inferences. If, on the other hand, the prediction is to be disconfirmed, then this would suggest that either the BIP model fails as an account of human goal inferences, or some constraint other than the ones we considered also suffices to render the BIP model tractable. The latter option may then be one that BIP modelers find interesting to pursue further.

Note that our predictions are only valid for those situations that agree with the simplifying assumptions of the model. Even though the simplifying assumptions may not hold in general, the predictions can be empirically tested under laboratory conditions that do agree with these assumptions. Doing so, however, will require experimental paradigms that differ in important respects from the paradigms used by Baker et al. (2009, 2007), in which actions are limited to be 'changes in locations' in a (2-dimensional) Euclidean space and goals are limited to be 'locations' in that space. The reason is that our predictions critically depend on there being the possibility that the observed agent has multiple simultaneous goals. Given that an agent cannot possibly be in two distinct locations at the same time, an agent cannot rationally have this as simultaneous goals. The testing of our predictions will hence require a paradigm in which the observed agent can reasonably be attributed goals other than 'being at certain locations' alone. The design of such a paradigm seems to us non-trivial, and certainly beyond the scope of this paper, but we hope that our predictions may motivate the design of such paradigms in the future.

6.3. Relaxing model assumptions

The BIP model of inverse planning that we analyzed includes several simplifying assumptions that may affect computational complexity results. We will briefly state some of these assumptions and reflect on the extent to which the computational complexity of the model depends on these assumptions. For some assumptions, breaking them will have no impact on the computational complexity. This means that the proofs and results in this paper are applicable to new model variants that do not make these simplifying assumptions. For others, it does have an impact. If one breaks such an assumption to achieve better model validity, then one has to re-analyze the computational complexity to explain when and why quick goal inference is possible.

The BIP model includes the assumption that actions and states are represented by single variables. One might want to break this assumption, because actions and states in the real world often have internal structure that is not reflected in such an atomic representation, i.e., a single value for a single variable. For instance, in the 'mother–son' example from Section 3 a state could (at the very least) represent the son's location, whether or not his stomach is rumbling, he has something in his hands, he is wearing his favorite shirt, time of the day, if he just ate dinner, etc. Each of these dimensions may be probabilistically dependent or independent. Yet, given that in the BIP model all actions and states are observed, any (in)dependence relations that may exist between them do not affect the inference of the most likely goals. Hence our complexity results remain applicable even for (observed) states and actions with internal structure.

This, however, brings us to another assumption of our BIP model: All states and actions are observed by the observer. While complete observability might be true in some situations (such as simplified experimental set-ups), in the real world parts of behavior are often unobservable. For example, actions and states can be unobserved because the observer blinks or because the actor moves into another room or because actions are occluded by an obstacle. In general, computing the most probable explanation for partially observed Bayesian networks (formally called MAXIMUM A POSTERIORI PROBABILITY) is computationally much harder than it is with full observability (Park & Darwiche, 2004). Given that our proofs for the tractability results depend heavily on the assumption of complete observability, these results do not necessarily generalize to situations that involve partial observability. Hence, the current analyses cannot yet explain how humans can quickly infer goals with partial observations.

Another form of unobservability arises when observers also attribute beliefs to actors. Baker, Saxe, and Tenenbaum (2011) have modeled a Bayesian theory of mind which incorporates beliefs. Their work, however, does not explicitly deal with the issue of (in)tractability. Furthermore, because the proofs in our paper depend heavily on the absence of such unobservable states, our analyses do not yet explain how people can quickly infer goals from actions under such conditions. Therefore such models may also benefit from an analysis of the kind that we presented in this article.

The final assumption that we discuss here is that goals are represented by single variables, without any internal structure. Such simple goal representations seem implausible for reasons similar to those stated above for simple action and state representations. To illustrate, consider again the son in our running example. His goal 'to satisfy big hunger' may be further decomposed into a set of interconnected (sub)goals, such as, 'go to kitchen', 'open cabinet door', 'get candy bar', etc. (cf. Baker et al., 2009, 2007; Uithol, van Rooij, Bekkering, & Haselager, 2012). Adjusting the BIP model to incorporate goals with such internal structure would not affect the applicability of the intractability results in this paper. Without any further constraints on the dependencies between (sub)goals, that model would remain intractable under the same conditions that we analyzed. Depending on the exact operationalization of complex goals the tractability results, however, are not guaranteed to carry over. Again, those models may also benefit from an analysis of the kind that we presented in this article.

7. Conclusion

In closing, we remark that our approach can be seen as exemplary of a general strategy for dealing with intractability in models of cognition. Our approach reveals that—contrary to popular belief—optimal Bayesian models can scale to complex, real-world domains *and* still explain when and why quick goal inferences are possible. To achieve this, Bayesian modelers need only identify constraints that apply in the real-world and suffice to render their models' computations tractable. By restricting Bayesian models in this way these models also become better testable: the constraints required to guarantee tractability of the models yield new predictions (specifically, about the speed and/or accuracy of participants) that can be used to perform more stringent tests of these models. Furthermore, this strategy of dealing with intractability of computational models of cognition differs from more common approaches such as postulating heuristics or approximations at the algorithmic level. Using complexity-theoretic analyses one can identify situational properties that, when constrained, render the computations postulated by the model tractable, in effect, explaining the speed of goal inference at the computational level.

Acknowledgments

We thank John Kruschke and four anonymous reviewers for their valuable comments on earlier versions of this paper.

Preliminary versions of parts of this article have appeared in the Proceedings of the 32nd *Annual Conference of the Cognitive Science Society* and have been presented at the 44th *Annual Meeting of the Society for Mathematical Psychology* (SMP). Mark Blokpoel was supported by a DCC Ph.D. grant awarded to Iris

van Rooij, Ivan Toni and Pim Haselager. Johan Kwisthout was supported by the OCTOPUS project under the responsibility of the Embedded Systems Institute. Todd Wareham was supported by NSERC Personal Discovery Grant #228104.

Appendix. Details of proofs

This appendix contains two fully detailed proofs referred to in Section 5. For completeness some definitions are repeated and we note that the numbering of theorems, corollaries and results in the appendix matches the numbering of the corresponding theorems, corollaries and results in the main paper.

A.1. Classical computational complexity analysis

To prove that BIP is not solvable in polynomial time we use Lemma A and prove that the model's decision variant is NP-hard. We first define D-BIP based on the value function of computing the marginal probability $\Pr(\mathbf{g}, \mathbf{s}, \mathbf{a})$. Computing the marginal probability is computable in polynomial time (van der Gaag, 1990). Note that using the marginal probability deviates from the original definition of BIP which is based on conditional probability $\Pr(\mathbf{g} \mid \mathbf{s}, \mathbf{a})$. However, maximizing the conditional probability over $\mathbf{g}$ is the same as maximizing the marginal probability over $\mathbf{g}$, because $\Pr(\mathbf{s}, \mathbf{a})$ is constant and $\Pr(\mathbf{g} \mid \mathbf{s}, \mathbf{a}) = \frac{\Pr(\mathbf{g}, \mathbf{s}, \mathbf{a})}{\Pr(\mathbf{s}, \mathbf{a})}$.

DECISION-BIP (D-BIP)
Input: Similar to BIP plus an integer $0 \leq q < 1$.
Question: Does there exist a joint value assignment $\mathbf{g}$ to all goal variables in $\mathbf{G}$ given $\mathbf{s} \cup \mathbf{a}$, such that $\Pr(\mathbf{g}, \mathbf{s}, \mathbf{a}) > q$?

To prove D-BIP is NP-hard we construct a polynomial time reduction from D-CLIQUE to D-BIP according to the blueprint in Box 1. This proves D-BIP is NP-hard, because D-CLIQUE is NP-hard Garey and Johnson (1979).

DECISION-CLIQUE (D-CLIQUE)
Input: A undirected graph $G = (V, E)$ where V is ordered and $k \in \mathbb{N} > 0$.
Question: Does there exist a subset $V' \subseteq V$ such that $|V'| = k$ and $\forall_{u,v \in V'}[(u, v) \in E]$?

Theorem A. D-BIP is NP-hard.

Proof. The following polynomial time reduction proves that D-BIP is NP-hard, because D-CLIQUE is NP-hard. We assume, without loss of generality, that D-CLIQUE instances have $k > 1$ and non-empty graphs.[10]

Step 1. Given an instance $\langle \mathcal{G} = (V, E), k \rangle$ of D-CLIQUE, translate it to an instance $\langle \mathcal{B}, \mathbf{a}, \mathbf{s}, q \rangle$ of D-BIP as follows:

i. Assume an arbitrary order on the vertices in V such that $V_1 < V_2 < \cdots < V_{|V|}$.
ii. Assume the basic structure in $\mathcal{B}$ as defined in D-BIP and create $1 + (k-1) + k(k-1)/2$ Boolean state variables $S_1, \ldots, S_{1+(k-1)+k(k-1)/2}$, $(k-1) + k(k-1)/2$ Boolean action variables $A_1, \ldots, A_{(k-1)+k(k-1)/2}$, and k goal variables $G_1, \ldots, G_k$, where $\Omega(G_i) = \{1, \ldots, |V|\}$. Now define $v : \mathbf{G} \rightarrow V$, where $v(G_i)$ returns V_j where $j = g_i$ if and only if there is a value g_i assigned to G_i.
iii. Set $\Omega(S_1) = \{true\}$ and for $2 \leq i \leq 1+(k-1)+k(k-1)/2$, let S_i depend on S_{i-1} and A_{i-1} and have the associated conditional probability

$$\Pr(S_i = true \mid S_{i-1}, A_{i-1}) = \begin{cases} 1 & \text{if } S_{i-1} = true \text{ and} \\ & A_{i-1} = true \\ 0 & \text{otherwise.} \end{cases}$$

These state variables effectively function as conjunctions which ensure that there is some assignment $\mathbf{g}$ to $\mathbf{G}$ such that $\Pr(\mathbf{g}) > 0$ if and only if all action variables are set to *true*.
iv. For $1 \leq i < k$, let A_i depend on G_i, G_{i+1}, and S_i and have the associated conditional probability

$$\Pr(A_i = true \mid G_i, G_{i+1}, S_i) = \begin{cases} 1 & \text{if } v(G_i) < v(G_{i+1}) \\ & \text{and } S_i = true \\ 0 & \text{otherwise.} \end{cases}$$

These action variables ensure that the values of the goal variables in any assignment $\mathbf{g}$ to $\mathbf{G}$ such that $\Pr(\mathbf{g}) > 0$ are distinct.
v. For $k \leq i \leq (k-1) + k(k-1)/2$, let each A_i depend on S_i and distinctly depend on a pair of goal variables (G_p, G_q), where $p \neq q$, and have the associated conditional probability

$$\Pr(A_i = true \mid G_p, G_q, S_i) = \begin{cases} 1 & \text{if } (v(G_p), v(G_q)) \in E \\ & \text{and } S_i = true \\ 0 & \text{otherwise.} \end{cases}$$

These action variables ensure that each pair of vertices in the set of vertices in $\mathcal{G}$ corresponding to the values of the goal variables in any assignment $\mathbf{g}$ to $\mathbf{G}$ such that $\Pr(\mathbf{g}) > 0$ is connected by an edge in $\mathcal{G}$.
vi. For $1 \leq i \leq (k-1) + k(k-1)/2$, omit all dependencies between A_i and any goal G_j not defined above.
vii. Let all state and action variables above be observed to be *true*, $q = 0$ and the prior probability distribution for each goal variable be uniform.

To prove that the construction above is a reduction, we must show that the answer to the given instance of D-CLIQUE is *yes* if and only if the answer to the constructed instance of D-BIP is *yes*, this corresponds to Steps 2 and 3.

Step 2. If the output for the given instance of D-CLIQUE is *yes*, then there exists at least one subset $V' \subseteq V$ such that $|V'| = k$ and for all $u, v \in V'$, $(u, v) \in E$. The fact that V' is a clique means that all elements in V' are unique and there is an edge between each pair of distinct vertices in V'. Note that only joint value assignments with these properties have probability $\Pr(\mathbf{g}, \mathbf{s}, \mathbf{a}) > 0$ in the constructed instance of D-BIP, due to the nodes created in Step 1.iv that enforce that the variables in $\mathbf{g}$ encode k distinct variables, and the nodes created in Step 1.v that enforce that all these variables are connected. Every other joint value assignment has probability $\Pr(\mathbf{g}, \mathbf{s}, \mathbf{a}) = 0$. Thus, if there is a k-clique in G, there exists at least one joint value assignment $\mathbf{g}$ with probability $p > 0$, namely the joint value assignments that encode these cliques in $\mathbf{g}$. Given the structure of $\mathcal{B}$, this implies that $\Pr(\mathbf{g}, \mathbf{s}, \mathbf{a}) > 0$ and thus that the output of the constructed instance of D-BIP is *yes*.

Step 3. If the answer to the constructed instance of D-BIP is *yes*, then there is an assignment $\mathbf{g}$ to $\mathbf{G}$ such that $\Pr(\mathbf{g}) > 0$. Given the structure of $\mathcal{B}$ and given that all action and state variables are observed to be *true*, the only possible joint value assignment $\mathbf{g}$ is the one in which the values of the goal variables are not only distinct, but also where these values correspond to a set of vertices such that every distinct pair of vertices encoded in $\mathbf{g}$ is connected by an edge in $\mathcal{G}$. Hence, the set of vertices corresponding to the goal

[10] If $k = 1$ and the graph is non-empty, then the instance is trivially a yes-instance. If the graph is empty, then the instance is trivially a no-instance. For both, one could transform them into a trivial yes- respectively no-instance of BIP.

variable values in $\mathbf{g}$ corresponds to a k-clique in $\mathcal{G}$, which means that the answer to the given instance of D-CLIQUE is *yes*.

Step 4. As the number of conditional probability tables that need to be constructed above is proportional to the total number of variables (which is $|\mathbf{S}|+|\mathbf{A}|+|\mathbf{G}| = (1+(k-1)+k(k-1)/2)+((k-1)+k(k-1)/2)+k$) and each table involves at most 3 variables with at most $|V|$ values per variables (giving tables with at most $|V|^3$ entries), this construction can be done in time polynomial in the size of the given instance of D-CLIQUE. □

Result 1. BIP is not computable in polynomial time, unless P = NP.

Proof. Follows from Theorem A, Lemma A and polynomial time computability of $\Pr(\mathbf{g}, \mathbf{s}, \mathbf{a})$. □

A.2. Parameterized computational complexity analysis

The following proof consists of a reduction from $\{k\}$-D-CLIQUE to $\{|\mathbf{A}|, s, a, g\}$-D-BIP to prove $\{|\mathbf{A}|, s, a, g\}$-D-BIP is W[1]-hard. Again we assume, without loss of generality, that D-CLIQUE instances have $k > 1$ and non-empty graphs.

Corollary B. $\{|\mathbf{A}|, s, a, g\}$-D-BIP is W[1]-hard.

Proof. *Step* 1. Given an instance $\langle \mathcal{G} = (V, E), k\rangle$ of k-D-CLIQUE, translate it to an instance $\langle \mathcal{B}, \mathbf{a}, \mathbf{s}, q\rangle$ of $\{|\mathbf{A}|, s, a, g\}$-D-BIP similar to the transformation in Step 1 in the proof of Theorem A, but change the following:

Instead of k goal variables, create k blocks of $\lceil \log_2 |V| \rceil$ ordered Boolean goal variables. We use these blocks of Boolean goal variables $\mathbf{B} = \mathbf{B_1}, \ldots, \mathbf{B_k}$ to encode the vertices of the clique instance. We change the definition of v to $v : \mathbf{B} \to V$, such that $v(\mathbf{B_i})$ returns V_j, where j is the number between 1 and $|V|$ encoded in binary in the values of the $\lceil \log_2 |V| \rceil$ ordered Boolean goal variables of $\mathbf{B_i}$.

For $1 \leq i < k$, let A_i depend on $\mathbf{B_i}$, $\mathbf{B_{i+1}}$, and S_i and have the associated conditional probability

$$\Pr(A_i = \mathit{true} \mid \mathbf{B_i}, \mathbf{B_{i+1}}, S_i) = \begin{cases} 1 & \text{if } v(\mathbf{B_i}) < v(\mathbf{B_{i+1}}) \\ & \text{and } S_i = \mathit{true} \\ 0 & \text{otherwise.} \end{cases}$$

For $1 \leq i < k$, let A_i instead depend on $\mathbf{B_i}$, $\mathbf{B_{i+1}}$, and S_i and have the associated conditional probability

$$\Pr(A_i = \mathit{true} \mid \mathbf{B_i}, \mathbf{B_{i+1}}, S_i) = \begin{cases} 1 & \text{if } v(\mathbf{B_i}) < v(\mathbf{B_{i+1}}) \\ & \text{and } S_i = \mathit{true} \\ 0 & \text{otherwise.} \end{cases}$$

Steps 2 & 3. These steps are the same as Steps 2 and 3 in the proof of Theorem A, modulo the replacement of multi-value goal variables by ordered sets of Boolean goal variables. This is, however, solved by the new definition of $v : \mathbf{B} \to V$.

Step 4. The transformation in Step 1 runs in fixed parameter tractable time, as the number of variables is a function of parameter k of D-CLIQUE: $|\mathbf{S}|+|\mathbf{A}|+|\mathbf{G_C}| = (1+(k-1)+k(k-1)/2)+((k-1)+k(k-1)/2)+k\lceil \log_2 |V| \rceil$. Also each conditional probability table involves at most $3\lceil \log_2 |V| \rceil$ Boolean variables (and hence has at most $2^{3\lceil \log_2 |V| \rceil} = (2^{\lceil \log_2 |V| \rceil})^3 \leq (2^{1+\log_2 |V|})^3 = (2|V|)^3 = 8|V|^3$ table entries).

Step 5. The transformation ensures that $|\mathbf{A}| = 1 + (k-1) + k(k-1)/2 + 1$ and $s = a = g = 2$. □

From Corollary B and Lemma B we can now conclude the computational complexity of BIP with respect to the parameters in this set.

Result 3. BIP is not fp-tractable time for the parameter set $\{|\mathbf{A}|, s, a, g\}$, unless FPT = W[1].

References

Aaronson, S. (2005). NP-complete problems and physical reality. *ACM SIGACT News*, *36*, 30–52.

Abdelbar, A. M., & Hedetniemi, S. M. (1998). Approximating MAPs for belief networks is NP-hard and other theorems. *Artificial Intelligence*, *102*, 21–38.

Arora, S. (1998). The approximability of NP-hard problems. In *Proceedings of the thirtieth annual ACM symposium on theory of computing* (pp. 337–348).

Baker, C. L., Saxe, R., & Tenenbaum, J. B. (2009). Action understanding as inverse planning. *Cognition*, *113*, 329–349.

Baker, C. L., Saxe, R., & Tenenbaum, J. B. (2011). Bayesian theory of mind: modeling joint belief–desire attribution. In L. Carlson, C. Hoelscher, & T. Shipley (Eds.), *Proceedings of the 33rd annual conference of the cognitive science society* (pp. 2469–2474). Austin, TX: Cognitive Science Society.

Baker, C. L., Tenenbaum, J. B., & Saxe, R. (2007). Goal inference as inverse planning. In D. McNamara, & J. Trafton (Eds.), *Proceedings of the 29th annual conference of the cognitive science society* (pp. 779–784). New York, NY: Lawrence Erlbaum Associates.

Baldwin, D. A., & Baird, J. A. (2001). Discerning intentions in dynamic human action. *Trends in Cognitive Sciences*, *5*, 171–178.

Blokpoel, M., Kwisthout, J., van der Weide, T. P., & van Rooij, I. (2010). How action understanding can be rational, Bayesian and tractable. In S. Ohlsson, & R. Catrambone (Eds.), *Proceedings of the 32nd annual conference of the cognitive science society* (pp. 1643–1648). Austin, TX: Cognitive Science Society.

Bodlaender, H. L., van den Eijkhof, F., & van der Gaag, L. C. (2002). On the complexity of the MPA problem in probabilistic networks. In F. van Harmelen (Ed.), *Proceedings 15th European conference on artificial intelligence* (pp. 675–679). Amsterdam, The Netherlands: IOS Press.

Bylander, T., Allemang, D., Tanner, M. C., & Josephson, J. R. (1991). The computational complexity of abduction. *Artificial Intelligence*, *49*, 25–60.

Charniak, E., & Goldman, R. P. (1993). A Bayesian model of plan recognition. *Artificial Intelligence*, *64*, 53–79.

Chater, N., Tenenbaum, J. B., & Yuille, A. (2006). Probabilistic models of cognition: conceptual foundations. *Trends in Cognitive Sciences*, *10*, 287–291.

Csibra, G., & Gergely, G. (2007). Obsessed with goals: functions and mechanisms of teleological interpretation of actions in humans. *Acta Psychologica*, *124*, 60–78.

Csibra, G., Gergely, G., Biró, S., Koós, O., & Brockbank, M. (1999). Goal attribution without agency cues: the perception of 'pure reason' in infancy. *Cognition*, *72*, 237–267.

Cuijpers, R. H., van Schie, H. T., Koppen, M., Erlhagen, W., & Bekkering, H. (2006). Goals and means in action observation: a computational approach. *Neural Networks*, *19*, 311–322.

Cummins, R. (2000). *"How does it work?" versus "What are the laws?": two conceptions of psychological explanation*. Cambridge, MA: The MIT Press.

Dennett, D. C. (1987). *The intentional stance*. Cambridge, MA: The MIT Press.

Downey, R., & Fellows, M. (1999). *Parameterized complexity*. Berlin: Springer-Verlag.

Fodor, J. (2001). *The mind doesn't work that way: the scope and limits of computational psychology*. Cambridge, MA: The MIT Press.

Fortnow, L. (2009). The status of the *P* versus *NP* problem. *Communications of the ACM*, *52*, 78–86.

Garey, M. R., & Johnson, D. S. (1979). *Computers and intractability: a guide to the theory of NP-completeness*. San Francisco, CA: W. H. Freeman.

Gergely, G., Nádasdy, Z., Csibra, G., & Biró, S. (1995). Taking the intentional stance at 12 months of age. *Cognition*, *56*, 165–193.

Ghahramani, Z. (1998). Learning dynamic Bayesian networks. In C. Giles, & M. Gori (Eds.), *Adaptive processing of sequences and data structures* (pp. 168–197). Berlin: Springer-Verlag.

Gigerenzer, G., Hoffrage, U., & Goldstein, D. G. (2008). Fast and frugal heuristics are plausible models of congition: reply to Dougherty, Franco–Watkins, and Thomas (2008). *Psychological Review*, *115*, 230–239.

Glass, D. H. (2007). Coherence measures and inference to the best explanation. *Synthese*, *157*, 275–296.

Haselager, W., van Dijk, J., & van Rooij, I. (2008). A lazy brain? Embodied embedded cognition and cognitive neuroscience. In P. Calvo, & T. Gomila (Eds.), *Handbook of embodied cognitive science: an embodied approach* (pp. 273–290). Oxford: Elsevier.

Hassin, R., Aarts, H., & Ferguson, M. J. (2005). Automatic goal inferences. *Journal of Experimental Social Psychology*, *41*, 129–140.

Jensen, F. V., & Nielsen, T. D. (2007). *Bayesian networks and decision graphs* (2nd ed.). New York: Springer-Verlag.

Király, I., Jovanovic, B., & Prinz, W. (2003). The early origins of goal attribution in infancy. *Consciousness and Cognition*, *12*, 752–769.

Kwisthout, J. (2010). Two new notions of abduction in Bayesian networks. In P. Bouvry, L. van der Torre, E. Dubois, & T. Latour (Eds.), *Proceedings of the 22nd Benelux conference on artificial tntelligence*, BNAIC 2010 (pp. 82–89).

Kwisthout, J. (2011). Most probable explanations in Bayesian networks: complexity and tractability. *International Journal of Approximate Reasoning*, *52*, 1452–1469.

Kwisthout, J., & van Rooij, I. (2013). Bridging the gap between theory and practice of approximate Bayesian inference. *Cognitive Systems Research*, *24*, 2–8.

Kwisthout, J., Wareham, H. T., & van Rooij, I. (2011). Bayesian intractability is not an ailment that approximation can cure. *Cognitive Science*, *35*, 779–784.

Marr, D. (1982). *Vision: a computational investigation into the human representation and processing of visual information/David Marr*. San Francisco, CA: W.H. Freeman.

Park, J. D., & Darwiche, A. (2004). Complexity results and approximation settings for MAP explanations. *Journal of Artificial Intelligence Research*, *21*, 101–133.

Pearl, J. (1988). *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Palo Alto: Morgan Kaufmann.
Pizlo, Z. (2008). *3D shape: its unique place in visual perception*. Cambridge, MA: The MIT Press.
Pylyshyn, Z. (1987). *The robot's dilemma: the frame problem in artificial intelligence*. Norwood, NJ: Ablex Publishing Corporation.
Robertson, N., & Seymour, P. (1986). Graph minors II: algorithmic aspects of tree-width. *Journal of Algorithms*, *7*, 309–322.
Sanborn, A., Griffiths, T., & Navarro, D. (2010). Rational approximations to rational models: alternative algorithms for category learning. *Psychological Review*, *117*, 1144–1167.
Shimony, S. E. (1994). Finding MAPs for belief networks is NP-hard. *Artificial Intelligence*, *68*, 399–410.
Thagard, P., & Verbeurgt, K. (1998). Coherence as constraint satisfaction. *Cognitive Science*, *22*, 1–24.
Uithol, S., van Rooij, I., Bekkering, H., & Haselager, P. (2011). What do mirror neurons mirror? *Philosophical Psychology*, *24*, 607–623.
Uithol, S., van Rooij, I., Bekkering, H., & Haselager, P. (2012). Hierarchies in action and motor control. *Journal of Cognitive Neuroscience*, *24*, 1077–1086.
van der Gaag, L. 1990. Probability-based models for plausible reasoning. Ph.D. Thesis, University of Amsterdam.
van Rooij, I. (2008). The tractable cognition thesis. *Cognitive Science*, *32*, 939–984.
van Rooij, I., Evans, P., Müller, M., Gedge, J., & Wareham, T. (2008). Identifying sources of intractability in cognitive models: an illustration using analogical structure mapping. In B. Love, K. McRae, & V. Sloutsky (Eds.), *Proceedings of the 30th annual conference of the cognitive science society* (pp. 915–920). Austin, TX: Cognitive Science Society.
van Rooij, I., Haselager, W., & Bekkering, H. (2008). Goals are not implied by actions, but inferred from actions and contexts. *Behavioral and Brain Sciences*, *31*, 38–39.
van Rooij, I., & Wareham, T. (2012). Intractability and approximation of optimization theories of cognition. *Journal of Mathematical Psychology*, *56*, 232–247.
van Rooij, I., Wright, C., & Wareham, T. (2012). Intractability and the use of heuristics in psychological explanations. *Synthese*, *187*, 471–487.