ILP-based Reasoning for Weighted Abduction

Naoya Inoue and **Kentaro Inui**
Graduate School of Information Sciences
Tohoku University
Aramaki Aza Aoba 09, Aoba-ku
Sendai 980-8579, Japan
{naoya-i, inui}@ecei.tohoku.ac.jp

Abstract

Abduction is widely used in the task of plan recognition, since it can be viewed as the task of finding the best explanation for a set of observations. The major drawback of abduction is its computational complexity. The task of abductive reasoning quickly becomes intractable as the background knowledge is increased. Recent efforts in the field of computational linguistics have enriched computational resources for commonsense reasoning. The enriched knowledge base facilitates exploring practical plan recognition models in an open-domain. Therefore, it is essential to develop an efficient framework for such large-scale processing. In this paper, we propose an efficient implementation of Weighted abduction. Our framework transforms the problem of explanation finding in Weighted abduction into a linear programming problem. Our experiments showed that our approach efficiently solved problems of plan recognition and outperforms state-of-the-art tool for Weighted abduction.

1 Introduction

An agent's beliefs and intention to achieve a goal is called a *plan*. *Plan recognition*, is thus to infer an agent's plan from observed actions or utterances. Recognizing plans is essential to natural language processing (NLP) tasks (e.g., story understanding, dialogue planning) as well as to acquire richer world knowledge. In the NLP research field, computational models for plan recognition have been studied extensively in the 1980s and 1990s (Allen and Perrault 1980; Carberry 1990; Charniak and Goldman 1991; Ng and Mooney 1992; Charniak and Shimony 1994, etc.). Yet, the models have not been tested on open data since the researchers suffered from a shortage of world knowledge, and hence it has not been demonstrated that they are robust. In the several decades since, however, a number of methods for large-scale knowledge acquisition have been proposed (Suchanek, Kasneci, and Weikum 2007; Chambers and Jurafsky 2009; Poon and Domingos 2010; Schoenmackers et al. 2010, etc.), and the products of their efforts have been made available to the public. Now we are able to tackle the problem of plan recognition *in an open-domain*.

Copyright © 2011, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

Since the task of plan recognition can be viewed as finding the best explanation (i.e., a plan) for an observation (i.e., utterances), most of the proposed methods have been based on *abduction*, the inference process of generating hypotheses to explain observations using background knowledge. It is crucial to use large-scale background knowledge to perform abductive inference in a wider domain. However, as the background knowledge is increased, the task of abductive reasoning quickly becomes intractable (Blythe et al. 2011; Ovchinnikova et al. 2011, etc.). Since most of models that have been proposed up to the present have not been designed for use with large-scale knowledge bases, we cannot receive the full benefits of large-scale processing.

In this paper, we propose an efficient framework of abduction that finds the best explanation by using the Integer Linear Programming (ILP) technique. Our system converts a problem of abduction into an ILP problem, and solves the problem by using efficient existing techniques developed in the ILP research community. Since our framework is based on Hobbs et al. (1993)'s *weighted abduction*, our framework is capable of evaluating the goodness of hypotheses based on their costs.

The rest of this paper is organized as follows. In the next section, we briefly review abduction and previous work about abduction. In Section 2, we describe the framework of weighted abduction, and then propose ILP formulation for weighted abduction in Section 3. We then apply our models to the existing dataset and demonstrate our approach outperforms state-of-the-art tool for weighted abduction in Section 5. Finally, the conclusion is presented along with possibilities for further study.

2 Background

We briefly give a description of abductive inference, and then review earlier work on abduction.

2.1 Abduction

Abduction is inference to the best hypothesis to explain observations using background knowledge. Abduction is widely used for a system that requires finding an explanation to observations, such as diagnostic systems and plan recognition systems. Formally, logical abduction is usually defined as follows:

- **Given:** Background knowledge B, observations O, where both B and O are sets of first-order logical formulae.
- **Find:** A hypothesis H such that $H \cup B \models O, H \cup B \not\models \bot$, where H is also a set of first-order logical formulae.

Typically, B is restricted to a set of first-order Horn clauses, and both O and H are represented as an existentially quantified clause that has a form of a conjunction of ground positive literals. In general, there are a number of hypotheses H that explains O. We call each hypothesis H that explains O a *candidate hypothesis*, and a literal $h \in H$ as an *elemental hypothesis*. The goal of abduction is to find the best hypothesis among candidate hypotheses by a specific evaluation measure. We call the best hypothesis H^* the *solution hypothesis*. Earlier work has used a wide range of evaluation measures to find a solution hypothesis such as the number of elemental hypotheses, the cost of a hypothesis or probability of a hypothesis etc.

2.2 Related work

We review prior efforts on abduction in terms of two viewpoints: *evaluation measure* and *scalability*.

Evaluation measure Previous work on the framework of abductive inference can be grouped into two categories in terms of the evaluation measure of hypothesis: *cost-based* approaches and *probabilistic* approaches. In *cost-based* approaches (Charniak and Shimony 1994; Hobbs et al. 1993, etc.), the system tries to find a hypothesis that has a minimum cost among other competing hypotheses, and identifies it as the best hypothesis. Weighted abduction, which we adopted, belongs to this group. In *probabilistic* approaches (Poole 1993b; Charniak and Goldman 1991; Sato 1995; Charniak and Shimony 1994; Kate and Mooney 2009; Raghavan and Mooney 2010; Blythe et al. 2011, etc.), the system identifies the highest probability hypothesis as the best hypothesis. Charniak and Shimony (1994) demonstrated that an abductive inference model that finds the minimum-cost hypothesis is equivalent to one that finds the maximum-a-posteriori assignment over a belief network that represents the possible hypothesis space. However, to the best of our knowledge, Hobbs et al (1993)'s weighted abduction is only a framework that is shown to have the mechanism that quantifies the appropriateness of hypothesis specificity.

It is crucial to discuss how to evaluate the specificity of hypotheses. Traditionally, two extreme modes of abduction have been considered. The first is *most-specific abduction*. In most-specific abduction, what we can explain from background knowledge is all explained, which is suitable for diagnostic systems. Some cost-based approaches and probabilistic approach falls into this group (Charniak and Shimony 1994; Raghavan and Mooney 2010, etc.). The second is *least-specific abduction*. Literally, in this mode an explanation is just assuming observations. In some cases of natural language understanding, we need this mode. With respect to abduction for plan recognition, adopting only one of these levels is problematic. For example, if we adopt most-specific abduction, the plan recognition system yields too specific explanation such as *Bob took a gun because he would rob XYZ bank using a machine gun which he had bought three days ago*. Conversely, if we adopt least-specific abduction, the system assumes just observation, as in *Bob took a gun because he took a gun*. We thus want to determine the suitable specificity during inference. Therefore, we adopt Hobbs et al (1993)'s weighted abduction in order to represent the specificity of hypotheses.

Scalability To perform plan inference in a broader domain, the size of the knowledge base is a crucial point. However, as we increase its size, abductive inference models immediately suffer from the exponentially-growing computational cost (Blythe et al. 2011; Ovchinnikova et al. 2011). Many researchers have tried to avoid this problem by a range of methods from approximation (Poole 1993a; Ishizuka and Matsuo 1998, etc.) to exact inference (Santos 1994, etc.). Santos (1994) formalized cost-based abduction (Charniak and Shimony 1994) as a linear constraint satisfaction problem (LCSP), and efficiently obtained the best hypothesis by solving the LCSP with a linear programming (LP) technique. He converted propositions generated during abductive inference into LP variables, and used the sum-product of these variables and the costs as the LP objective function. Our approach also adopts LP formulation, and performs a similar translation. However, his approach is based on propositional logic, and is incapable of evaluating the specificity of a hypothesis. The comparison with our approach is more detailed in Section 4.2

3 Weighted abduction

Hobbs et al. (1993) propose the framework of text understanding based on the idea that interpreting sentences is to prove the logical form of the sentence. They demonstrated that a process of natural language understanding, such as word sense disambiguation or reference resolution, can be described in the single framework based on abduction.

As mentioned before, abduction needs to select the best hypothesis, and hence this framework also needs to select the best interpretation based on some evaluation measure. Hobbs et al. extended their framework so that it gives a cost to each interpretation as the evaluation measure, and chooses the minimum cost interpretation as the best interpretation. This framework is called *weighted abduction*. In weighted abduction, observations are given with costs, and background axioms are given with weights. It then performs backward-reasoning on each observation, propagates its cost to the assumed literals according to the weights on the applied axioms, and merges redundancies where possible. A cost of interpretation is then the sum of all the costs on elemental hypotheses in the interpretation. Finally, it chooses the lowest cost interpretation as the best interpretation.

3.1 The basics

Following (Hobbs et al. 1993), we use the following representations for background knowledge, observations, and hypothesis in weighted abduction:

- **Background knowledge** B: a set of first-order logical formulae whose literals in its antecedent are assigned pos-

itive real-valued *weights*. In addition, both antecedent and consequent consist of a conjunction of literals. We use a notation p^w to indicate "a literal p has the weight w."

- **Observations** O: an existentially quantified conjunction of literals. Each literal has a positive real-valued cost. We use a notation $p^{\$c}$ to denote "a literal p has the cost c," and $c(p)$ to denote "the cost of the literal p."
- **Hypothesis** H: an existentially quantified conjunction of literals. Each literal also has a positive real-valued cost. The cost of H is then defined as $c(H) = \sum_{h \in H} c(h)$.

In the Hobbs et al.'s framework, inference procedure is only defined on the formats defined above, although neither formats of B, O nor H are mentioned explicitly.

3.2 Procedure of weighted abductive inference

Like logical abduction, H is abductively inferred from O and B, and the costs of elemental hypotheses in H are passed back from O multiplying the weights on the applied axioms in B. When two elemental hypotheses are unified, the smaller cost is assigned to the unified literal. Let us illustrate how these procedure works taking the following axioms and observations as an example:

$$B = \{\forall x(p(x)^{0.3} \wedge q(x)^{0.9} \Rightarrow r(x)), \quad (1)$$
$$\forall x \exists y(p(y)^{1.3} \Rightarrow b(x)), \quad (2)$$
$$O = \exists a(r(a)^{\$20} \wedge b(a)^{\$10}) \quad (3)$$

A candidate hypothesis that immediately arises is simply assuming O, i.e., $H_1 = \exists a(r(a)^{\$20} \wedge b(a)^{\$10})$, where $c(H_1) = \$20 + \$10 = \$30$. If we perform backward inference on $r(a)^{\$20}$ using axiom (1), we get $H_2 = \exists a(p(a)^{\$6} \wedge q(a)^{\$18} \wedge b(a)^{\$10})$ and $c(H_2) = \$34$. As we said, the costs are passed back from $r(a)^{\$20}$ multiplying the weights on axiom (1), and hence $c(p(a)) = \$20 \cdot 0.3 = \$6$ and $c(q(a)) = \$20 \cdot 0.9 = \$18$.

If we perform backward inference on both $r(a)$ and $b(a)$ by using axiom (1) and (2), we get another candidate hypothesis $H_3 = \exists a, b(p(a)^{\$6} \wedge q(a)^{\$18} \wedge p(b)^{\$13})$, in which $p(a)^{\$6}$ is unifiable with $p(b)^{\$13}$ assuming that a and b to be identical. In weighted abduction, since the cost of unified literal is given by the smaller cost, H_3 is refined as $\exists b(q(b)^{\$18} \wedge p(a)^{\$6})$, and $c(H_3) = \$24$. Considering only these three candidate hypotheses, a solution hypothesis $H^* = H_3$, which has a minimum cost $c(H_3) = \$24$.

We mentioned that weighted abduction is able to evaluate the specificity of a hypothesis in Section 2.2. The mechanism of specificity evaluation is accomplished by the propagation of costs. We can see the working example of this mechanism in the toy problem above: comparing $c(H_1)$ with $c(H_2)$ means determining if $r(a)$ should be explained more specifically or not.

4 ILP-based reasoning for weighted abduction

Now we describe our approach to perform weighted abduction using ILP. Our approach, similar to a typical abductive inference system, takes a set of logical formulae as background knowledge, and a conjunction of ground literals as observations. Our system finally outputs an existentially quantified conjunction of literals as a solution hypothesis.

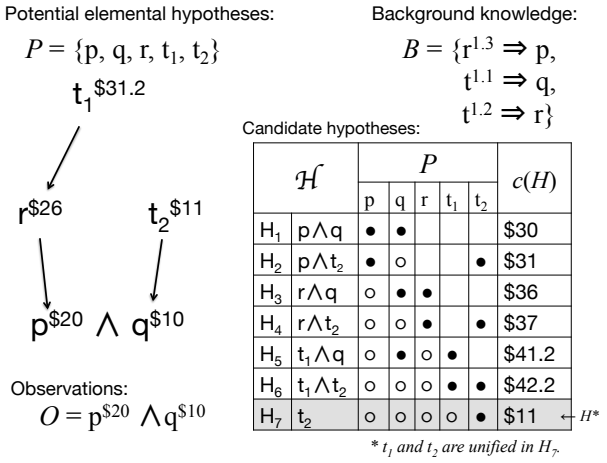


Figure 1: The combinatorial representation of candidate hypotheses by set P of potential elemental hypotheses. The black circle indicate that a proposition is in H_i, while the white circle indicate that a proposition is explained by $H_i \cup B$.

In this section, we first show candidate hypotheses in weighted abduction can be generated by applying three simple operations. Secondly, we then formulate weighted abduction as an optimization problem based on these operations.

4.1 Operations for hypotheses generation

Let B be background knowledge, O be observations and $\mathcal{H} = \{H_1, H_2, ...H_n\}$ be a set of candidate hypotheses, each of which is defined in Section 3.1. In order to enumerate candidate hypothesis H_i, we can execute the following three operations an arbitrary number of times (except *Initialization*).

Initialization

$$H \leftarrow O \quad (4)$$

Backward reasoning

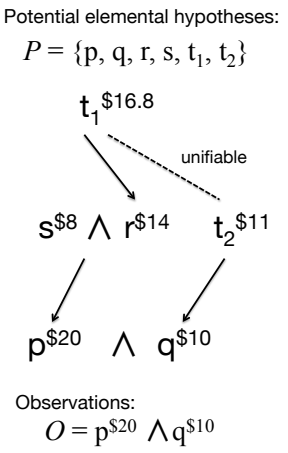
$$\frac{\bigwedge_{i=1}^{n} p_i^{w_i} \Rightarrow q \in B, q^{\$c_q} \subseteq H}{\bigwedge_{i=1}^{n} p^{\$w_i \cdot c_q}} \quad (5)$$

$$H \leftarrow H \wedge \bigwedge_{i=1}^{n} p_i^{\$w_i \cdot c_q} \quad (6)$$

Unification

$$\frac{p(X)^{\$c_x} \in H, p(Y)^{\$c_y} \in H, \exists \theta(p(X)\theta = p(Y)\theta)}{X = Y} \quad (7)$$

$$H \leftarrow H \setminus p(X)^{max(\$c_x, \$c_y)} \quad (8)$$



Background knowledge:

$B = \{s^{0.4} \wedge r^{0.7} \Rightarrow p,$
$t^{1.1} \Rightarrow q,$
$t^{1.2} \Rightarrow r\}$

Example of constraints:

C1: h_p=1, h_q=1
C2: $r_p \leq h_s + h_r$, $h_s = h_r \cdot 1$
$r_{t1} \leq u_{t1,t2}$
C3: $u_{t1,t2} \leq \frac{1}{2}(h_{t1} + h_{t2})$

Candidate hypotheses:

$\mathcal{H}$		P													$c(H)$
		h_p	r_p	h_q	r_q	h_r	r_r	h_s	r_s	h_{t1}	r_{t1}	h_{t2}	r_{t2}	$u_{t1,t2}$	
H_1	$p \wedge q$	1	0	1	0	0	0	0	0	0	0	0	0	0	\$30
H_2	$p \wedge t_2$	1	0	1	1	0	0	0	0	0	0	1	0	0	\$31
H_3	$s \wedge r \wedge q$	1	1	1	0	1	0	1	0	0	0	0	0	0	\$32
H_4	$s \wedge r \wedge t_2$	1	1	1	1	1	0	1	0	0	0	1	0	0	\$33
H_5	$s \wedge t_1 \wedge q$	1	1	1	0	1	1	1	0	1	0	0	0	0	\$34.8
H_6	$s \wedge t_1 \wedge t_2$	1	1	1	1	1	1	1	0	1	0	1	0	0	\$35.8
H_7	$s \wedge t_2$	1	1	1	1	1	1	1	0	1	1	1	0	1	\$19

Figure 2: ILP representation for the space of candidate hypotheses in the case for propositional logic

Our central idea of ILP formulation follows. Once we enumerate all elemental hypotheses that would be generated by operations above (henceforth we call *potential elemental hypotheses*), candidate hypotheses can be represented as an arbitrary combination of potential elemental hypotheses. We use P to denote a set of potential elemental hypotheses. This idea is illustrated in Figure 1. Firstly set P of potential elemental hypotheses is initialized by observation O and enumerated by backward reasoning on these hypotheses, and finally we get $P = \{p, q, r, t_1, t_2\}$. We give an unique assignment to each literal generated by backward chaining, since a hypothesis where unifiable literals are unified as in H_7 can be different from another where they are not as in H_6 in the case of predicate logic as a consequence of variable substitution. That is why we leave two literals t_1 and t_2 in P for the back-chained proposition t, and consider distinct two candidate hypotheses.

Based on this idea, it is quite easy to extend hypothesis finding to an optimization problem. For each $p \in P$, if we had a 0-1 state variable that represents whether or not the elemental hypothesis is included in a candidate hypothesis, as in Figure 1, all possible $H \in \mathcal{H}$ can be expressed as the combination of these state variables. Since our goal is to find a hypothesis that has a minimum cost, this representation is immediately used to formulate weighted abduction as an optimization problem which finds the assignment of state variables that minimizes the cost function. Note that the number of candidate hypotheses is $O(2^n)$, where n is the number of potential elemental hypotheses. We immediately see that the approach which finds a minimal hypothesis by evaluating all the candidate hypotheses intractable.

4.2 ILP-based formulation

First of all, we show how candidate hypotheses are expressed in ILP variables. We start with the simplest case, i.e., B, O and H are restricted to propositional logic formulae. We describe our ILP variables and constraints by using a toy problem illustrated in Figure 2.

Hypothesis inclusion We introduce an ILP variable $h \in \{0, 1\}$ defined as follows:

$$h_p = \begin{cases} 1 & \text{if } p \in H \text{ or } H \cup B \models p \\ 0 & \text{otherwise} \end{cases} \quad \text{for each } p \in P$$

For example, H_2 in Figure 2 holds $h_p = 1, h_q = 1$, where p is included in H_2, and q is explained by t_2 (i.e., $H_2 \cup B \models q$). Note that the state $h = 1$ is corresponding to the black circle and white circle in Figure 1.

Zero cost switching If we perform backward reasoning on elemental hypotheses, the back-chained literals are explained by the newly abduced literals, which means that these elemental hypotheses do not pay its cost any more. In addition, when two elemental hypotheses are unified, the bigger cost of the elemental hypothesis is excluded. This also implies that this elemental hypothesis does not pay its cost. We thus introduce an ILP variable $r \in \{0, 1\}$ defined as follows:

$$r_p = \begin{cases} 1 & \text{if } p \text{ does not pay its cost} \\ 0 & \text{otherwise} \end{cases} \quad \text{for each } p \in P$$

In Figure 2, r_q in H_2 is set to 1 since q is explained by t_2.

State of unification We prepare an ILP variable $u \in \{0, 1\}$ for expressing whether or not two elemental hypotheses $p \in P$ and $q \in P$ are unified:

$$u_{p,q} = \begin{cases} 1 & \text{if } p \text{ is unified with } q \\ 0 & \text{otherwise} \end{cases} \quad \text{for each } p, q \in P$$

In Figure 2, u_{t_1,t_2} in H_7 is set to 1 since t_1 and t_2 are unified.

Now that we can define $c(H)$ by the sum of the costs for $p \in P$ such that p is included in a candidate hypothesis (i.e., $h_p = 1$) and is *not* explained (i.e., $r_p = 0$), which is the objective function of our ILP problem:

$$\text{minimize } c(H) = \sum_{p \in \{p | p \in P, h_p = 1, r_p = 0\}} c(p), \qquad (9)$$

where $c(p)$ is the cost of a literal p passed back from observations according to *backward-reasoning* operation in Section 4.1 when all potential elemental hypotheses are enumerated in advance. However, a possible world represented by these ILP variables up to now includes an invalid candidate hypothesis (e.g., an elemental hypothesis might not pay its cost even though it is *neither* unified *nor* explained). Accordingly, we introduce constraints that limit a possible world in ILP representation to only valid hypothesis space.

Constraint 1 Observation literals are always included in or explained by a candidate hypothesis.

$$h_p = 1 \quad \text{for each} \quad p \in O \tag{10}$$

Constraint 2 An elemental hypothesis $p \in P$ does not have to pay its cost (i.e., $r_p = 1$) only if it is explained *or* unified. Namely, in order to set $r_p = 1$, at least one literal e such that explains p is included in or explained by a candidate hypothesis (i.e., h_e = 1), *or* p is unified with at least one literal q such that $c(q) < c(p)$ (i.e., $u_{p,q} = 1$). This can be expressed as the following inequality:

$$r_p \leq \sum_{e \in expl(p)} h_e + \sum_{q \in sml(p)} u_{p,q} \quad \text{for each} \quad p \in P, \tag{11}$$

where $expl(p) = \{e \mid e \in P, \{e\} \cup B \models p\}$, and $sml(p) = \{q \mid q \in P, c(q) < c(p)\}$. In Figure 2, $r_p \leq h_s + h_r$ is created to condition that q may not pay its cost only if q is explained by $s \wedge r$. The constraint for t_1, $r_{t_1} \leq u_{t_1,t_2}$, states that t_1 may not pay its cost only if it is unified with t_2. Note that this constraint is not generated for t_2 since $c(t_1) > c(t_2)$.

Furthermore, if literals $q_1, q_2, ..., q_i$ obtained by $expl(p)$ are the form of conjunction (i.e., $q_1 \wedge q_2 \wedge ... \wedge q_i$), we use an additional constraint to force their inclusion states are consistent with the others (i.e., $h_{q_1} = h_{q_2} = ... = h_{q_i}$). This can be expressed as the following inequality:

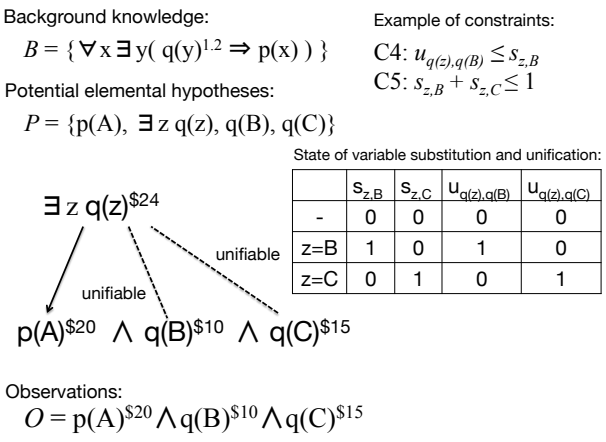
$$\sum_{a \in and(p)} h_a = h_p \cdot |and(p)| \quad \text{for each} \quad p \in P, \tag{12}$$

where $and(p)$ denotes a set of $a \in P$ such that a is conjoined with p by conjunction. In Figure 2, $h_s = h_r \cdot 1$ is generated to represent that s and r are literals conjoined by *logical and*. We need this constraint since inequality (11) allows reducing even when *one of* literals obtained by $expl(p)$ is included in or explained by a candidate hypothesis.

Constraint 3 Two elemental hypotheses $p, q \in P$ can be unified (i.e., $u_{p,q} = 1$) only if both p and q are included in or explained by a candidate hypothesis (i.e., $h_p = 1$ and $h_q = 1$).

$$u_{p,q} \leq \frac{1}{2}(h_p + h_q) \quad \text{for each} \quad p, q \in P \tag{13}$$

For example, in Figure 2, $u_{t_1,t_2} \leq \frac{1}{2}(h_{t_1} + h_{t_2})$ is generated for the condition of unification of t_1 and t_2.



	$s_{z,B}$	$s_{z,C}$	$u_{q(z),q(B)}$	$u_{q(z),q(C)}$
-	0	0	0	0
z=B	1	0	1	0
z=C	0	1	0	1

Figure 3: Unification in ILP framework for the case for first-order logic

Now we move on to the slightly more complicated case where first-order logic is used in B, O and H. The substantial difference from the case of propositional logic is that we must account for variable substitution to control the unification of elemental hypotheses. For example, if we observed $wife\_of(John, Mary) \wedge man(John)$ and had a knowledge $\forall x \exists y(wife\_of(x, y) \Rightarrow man(x))$, we could generate the potential elemental hypothesis $\exists z(wife\_of(John, z))$, where $John$ is a non-skolem constant, and z is existentially quantified variable. Then the hypothesis $\exists z(wife\_of(John, z))$ could only be unified with $wife\_of(John, Mary)$ if we assume $z = Mary$. In order to take variable substitution into account, we introduce new variables. Hereafter, we use V to denote a set of existentially quantified variables in P, and C to denote a set of non-skolem constants in P.[1] We use Figure 3 as an example.

Variable substitution When two literals are unified, a variable $x \in V$ in the literals is substituted for a constant or variable $y \in \{C \cup V\}$. We introduce the new variable $s \in \{0, 1\}$ defined as:

$$s_{x,y} = \begin{cases} 1 & \text{if } x \text{ is substituted for } y \\ 0 & \text{otherwise} \end{cases}$$

For example, $s_{z,C}$ in Figure 3 is set to 1 since the variable z is substituted for the constant C when $q(z)$ and $q(C)$ are unified.

We also use additional constraints that limits unification so that the framework checks the states of variable substitutions needed for the unification, and consistency of substitutions:

Constraint 4 Two literals $p, q \in P$ are allowed to be unified (i.e., $u_{p,q} = 1$) only when all variable substitutions x/y involved by the unification are activated (i.e.,

[1] Henceforth, we use the terms "variable" and "constant" to represent an existentially quantified variable and non-skolem constant for convenience.

$s_{x,y} = 1$).

$$u_{p,q} \leq \frac{\sum_{(x,y)\in usub(p,q)} s_{x,y}}{|usub(p,q)|} \text{ for each } p,q \in P, \quad (14)$$

where $usub(p,q)$ denotes a set of variable substitutions that are required to unify p and q. In Figure 3, the constraint $u_{q(z),q(B)} \leq s_{z,B}$ is generated since z needs to be substituted for B when $q(z)$ and $q(B)$ are unified.

Constraint 5 When a variable x can be substituted for multiple constants $A = \{a_1, a_2, ..., a_i\}$ (i.e., $s_{x,a_1} = 1, s_{x,a_2} = 1, ...,$ or $s_{x,a_i} = 1$), the variable x can be substituted for only one constant $a_i \in A$ (i.e., at most one s_{x,a_i} can be 1). This can be expressed as follows:

$$\sum_{a_i \in ucons(x)} s_{x,a_i} \leq 1 \text{ for each } x \in V, \quad (15)$$

where $ucons(x)$ denotes a set of constants which can be bound to a variable x. In Figure 3, since two constants B and C can be bound to the variable z, the constraint $s_{z,B} + s_{z,C} \leq 1$ is created.

Constraint 6 The binary relation over $(x,z) \in V \times \{V \cup C\}$ must be transitive (i.e., $s_{x,z}$ must be 1 if $s_{x,y} = 1$ and $s_{y,z} = 1$ for all $y \in V \cup C$). This can be expressed as the following constraints:

$$s_{x,y} + s_{y,z} \leq 2 \cdot s_{x,z} \quad (16)$$
$$s_{x,z} + s_{y,z} \leq 2 \cdot s_{x,y} \quad (17)$$
$$s_{x,y} + s_{x,z} \leq 2 \cdot s_{y,z} \quad (18)$$

Although this increases the number of constraints in $O(|V \cup C|^3)$, it is practical enough to consider the transitive relation over $(x,y,z) \in V_e \times V_e \times C_e$ in a cluster $e = \{V_e, C_e\} \in E$ formed by an equivalence class of such variables V_e and constants C_e for which are potentially substituted (i.e., possibly used to unify some literals), which typically produces more compact constraints. Moreover, considering the transitivity over clusters avoids the unnecessary growth of the number of constraints for two cases: a cluster has less than 2 constants. If a cluster had no constant, it would be unnecessary to enforce the transitivity since we would be able to regard all the variables in the cluster as one single unknown entity. Similarly, if a cluster had exactly one constant, all the variables in the cluster would be bound to the constant. Thus, we create the above three constraints for each $(x,y,z) \in V_e \times V_e \times C_e$ for each cluster $e \in \{e \mid e \in E, 2 \leq |C_e|\}$.

Our approach is different from Santos (1994)'s LP formulation in terms that our approach is capable of evaluating the specificity of hypotheses, as mentioned in Section 2.2. Specifically, explaining a literal p to reduce its cost (i.e., $r_p = 1$) by a literal q forces us to pay another cost for q instead (i.e., $h_q = 1$, see Constraint 2). Therefore, usually this new hypothesis q is meaning-less and is not favored since the cost of explanation does not change largely (i.e., *less* specific explanation is favored as in H_1 and H_3 in Figure 1). However, once we get a good hypothesis such that explains other hypotheses at the same time (i.e., unified with other literals), it is then favored as a result of drastic decrease of the explanation cost, as in H_7 in Figure 1 (i.e., *more* specific explanation is favored). In our framework, the specificity evaluation is successfully controlled by using the ILP variable h, r, u.

Another difference from Santos (1994)'s approach is that his approach does not take unification into account in inference process, while our work dynamically makes unification decision of literals. Our approach can express a literal involving undetermined arguments in B, O, H, and can evaluate the goodness of variable substitution during inference process at the same time. For example, suppose we have $B = \{\forall x, y(hate(x,y) \wedge hasBat(x) \Rightarrow hit(x,y))\}, O = \exists z(hit(z, Bob) \wedge hate(Mary, Bob) \wedge hate(Kate, Bob) \wedge hasBat(Kate))$. Back-chaining on $hit(z, Bob)$ yields the elemental hypotheses $\exists z(hate(z, Bob) \wedge hasBat(z))$, where we can consider two variable substitutions: $z = Mary$ or $z = Kate$. If we take $z = Mary$, we do not need to pay the cost for $hate(Mary, Bob)$ since it is unified with the observation. If we take $z = Kate$, we do not need to pay the costs for $hate(Kate, Bob)$ and $hasBat(Kate)$ since they are unified with the observations. Therefore, we want to choose the cost-less hypothesis that assumes the variable substitution $z = Kate$ as the best hypothesis. In our framework, this choice is efficiently expressed through the ILP variables u, s and their constraints.

5 Evaluation

We evaluated the efficiency of our ILP-based framework for weighted abduction by analyzing how the inference time changes as the complexity of abduction problems increases. In order to simulate the diversity of the complexity, we introduced a parameter for the setting of experiment d, which limits the depth of backward inference chain. If we set $d = 1$ and had p in observation, the framework would apply backward inference to p only once, i.e., it would not apply backward inference to the abduced literals p' any more. We also compared the performance with Mini-TACITUS[2] (Mulkar, Hobbs, and Hovy 2007), which is the state-of-the-art tool of weighted abduction. To the best of our knowledge, Mini-TACITUS is only a tool of weighted abduction available for now. We have investigated (i) how many problems in our testset Mini-TACITUS could solve in 120 seconds, and (ii) the average of its inference time for solved problems. For solving ILP, we have a range of choices from non-commercial solvers to commercial solvers. In our experiments, we adopted SCIP[3], which is the fastest solver among non-commercial solvers. SCIP solves ILP problems using the *branch-cut-and-price* method.

5.1 Dataset

Our test set was extracted from the dataset originally developed for Ng and Mooney (1992)'s abductive plan recognition system ACCEL. We extracted 50 plan recognition problems and 107 background axioms from the dataset. The plan recognition problems provide agents' partial actions as

[2] http://www.rutumulkar.com/

[3] http://scip.zib.de/

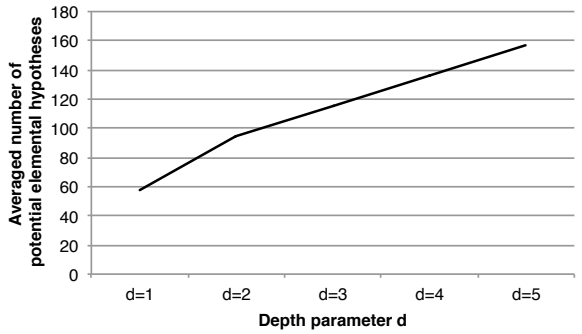


Figure 4: The complexity of each problem setting

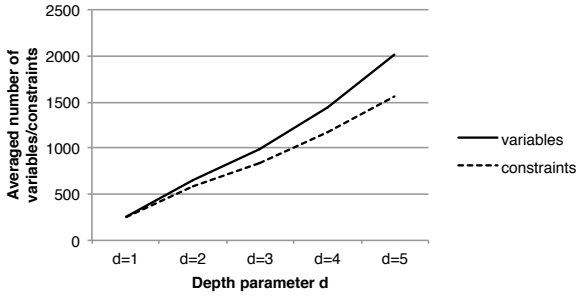


Figure 5: The complexity of each ILP problem

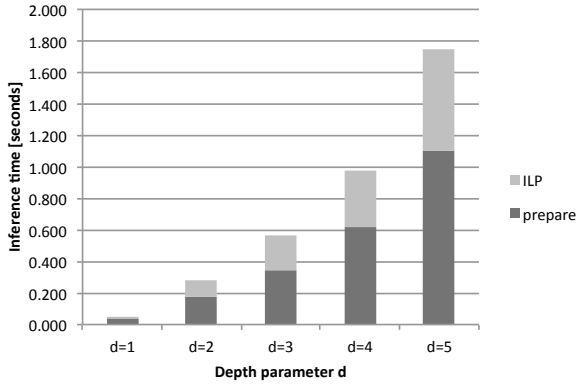


Figure 6: The efficiency of the ILP-based formulation "prepare" and "ILP" denote the time required to convert a weighted abduction problem to ILP problem, and the time required to solve the ILP problem respectively.

a conjunction of literals. For example, in the problem *t2*, the following observation literals are provided:

(1) $inst(get2, getting) \wedge agent\_get(get2, bob2) \wedge name(bob2, bob) \wedge patient\_get(get2, gun2) \wedge inst(gun2, gun) \wedge precede(get2, getoff2) \wedge inst(getoff2, getting\_off) \wedge agent\_get\_off(getoff2, bob2) \wedge patient\_get\_off(getoff2, bus2) \wedge inst(bus2, bus) \wedge place\_get\_off(getoff2, ls2) \wedge inst(ls2, liquor\_store)$

This logical form denotes a natural language sentence "*Bob got a gun. He got off the bus at the liquor store.*" The plan recognition system requires to infer *Bob*'s plan from these observations using background knowledge. The background knowledge base contains Horn-clause axioms such as:

(2) $inst(R, robbing) \wedge get\_weapon\_step(R, G) \Rightarrow inst(G, getting)$

From this dataset, we created two types of testsets: (i) *testset A*: Ng and Mooney's original dataset, (ii) *testset B*: a modified version of *testset A*. For both testsets, we assigned uniform weights to antecedents in background axioms so that the sum of those equals 1, and assigned \$20 to each observation literal. We created *testset B* so that the background knowledge base does not contain a constant in its arguments since Mini-TACITUS does not allow us to use constants in background knowledge axiom. Specifically, we converted the predicate $inst(X, Y)$ that denotes X is a instance of Y into a form of $inst\_Y(X)$ (e.g., $inst(get2, getting)$ is converted into $inst\_getting(get2)$). We also converted an axiom involving a constant in its arguments into *neo-Davidsonian* style. For example, $occupation(A, busdriver)$, where $busdriver$ is a constant, is converted to $busdriver(X) \wedge occupation(A, X)$. These two conversions did not affect the complexity of the problems substantially.

5.2 Results and discussion

We first show the complexity of abduction problems in our *testset A*. Figure 4 shows the number of potential elemental hypotheses, P described in Section 4.2, averaged for 50 problems. As d increases, the number of elemental hypotheses increases constantly. Recall that the number of candidate hypotheses is $O(2^n)$, where n is the number of potential elemental hypotheses. Therefore, in our testset, we roughly have $2^{160} \approx 1.46 \cdot 10^{48}$ candidate hypotheses for a propositional case if we set $d = 5$. Figure 5 illustrates the number of variables and constraints of a ILP problem for each parameter d, averaged for 50 problems. Although the complexity of the ILP problem increases, we can rely on an efficient algorithm to solve a complex ILP problem.

The results of inference time in our framework on *testset A* is given in Figure 6 in the two distinct measures: (i) the time of conversion to ILP problem, and (ii) the time ILP technique had took to find an optimal solution. Figure 6 demonstrates that our framework is capable of coping with larger scale problems, since the inference can be performed in polynomial time to the size of the problem.

Then we show the inference time of Mini-TACITUS on *testset B*. The complexity of the testset was quite similar to the *testset A* since the modification affecting the original complexity occurred in only 2 axioms. On *testset B*, we have confirmed that our framework had solved the 100% of the problems for $1 \leq d \leq 5$, and it took 1.16 seconds when averaged for the 50 problems of $d = 5$. The results

Table 1: The results of weighted abduction on Mini-TACITUS

	% of solved	Avg. of inference times [sec.]
$d = 1$	28.0% (14/50)	8.3
$d = 2$	20.0% (10/50)	10.2
$d = 3$	20.0% (10/50)	10.1

"% of solved" indicates that the ratio of problems Mini-TACITUS could solve in 120 seconds to all the 50 problems. "Avg. of inference times" denotes the inference time averaged for the solved problems.

of abductive reasoning on Mini-TACITUS is shown in Table 1. The results show that the 72% of the problems (36/50) could not be solved in 120 seconds for the easiest setting $d = 1$. For the slightly complex setting $d \geq 2$, 80% of the problems (40/50) could not be solved in 120 seconds. We found that no additional axioms were applied in the 10 solved problems for $d \geq 3$: the search space did not change. This indicates that Mini-TACITUS is sensitive to the depth parameter, which means the growth rate of inference time is very large. This becomes a significant drawback for abductive inference using large-scale background knowledge. Note that the inference time could not be directly compared with our results since our implementation is C++, whereas Mini-TACITUS is Java-based.

6 Conclusions

We have addressed the scalability issue of abductive reasoning. Since recent efforts on computational linguistics study have facilitated large-scale commonsense reasoning, the scalability issue is a significant problem. We have proposed an ILP-based framework for weighted abduction, which maps weighted abduction to a linear programming problem and efficiently finds an optimal solution. We have demonstrated that our approach efficiently solved the problems of abduction, and is promising compared with the state-of-the-art tool for weighted abduction. Future work includes extending our framework to use rich semantic representation in B, O and H. Our first plan is to incorporate a logical negation operator ($\neg$), which means that contradiction would also be expressed in our framework. The major advantage of an ILP-based framework is that most of the logical operators can be expressed through simple inequalities. Our future direction also includes incorporating forward chaining operation into our framework, where the entailment relation ($\Rightarrow$) is also represented as an inequality straightforwardly.

7 Acknowledgments

This work was supported by Grant-in-Aid for JSPS Fellows (22-9719).

References

Allen, J. F., and Perrault, C. R. 1980. Analyzing intention in utterances. *Artificial Intelligence* 15(3):143–178.

Blythe, J.; Hobbs, J. R.; Domingos, P.; Kate, R. J.; and Mooney, R. J. 2011. Implementing Weighted Abduction in Markov Logic. In *IWCS*.

Carberry, S. 1990. *Plan Recognition in Natural Language Dialogue*. MIT Press.

Chambers, N., and Jurafsky, D. 2009. Unsupervised Learning of Narrative Schemas and their Participants. In *ACL*, 602–610.

Charniak, E., and Goldman, R. P. 1991. A Probabilistic Model of Plan Recognition. In *AAAI*, 160–165.

Charniak, E., and Shimony, S. E. 1994. Cost-based abduction and map explanation. *Artificial Intelligence* 66(2):345–374.

Hobbs, J. R.; Stickel, M.; Appelt, D.; and Martin, P. 1993. Interpretation as Abduction. *Artificial Intelligence* 63:69–142.

Ishizuka, M., and Matsuo, Y. 1998. SL Method for Computing a Near-optimal Solution using Linear and Non-linear Programming in Cost-based Hypothetical Reasoning. In *PRCAI*, 611–625.

Kate, R. J., and Mooney, R. J. 2009. Probabilistic Abduction using Markov Logic Networks. In *PAIRS*.

Mulkar, R.; Hobbs, J. R.; and Hovy, E. 2007. Learning from Reading Syntactically Complex Biology Texts. In *The 8th International Symposium on Logical Formalizations of Commonsense Reasoning*.

Ng, H. T., and Mooney, R. J. 1992. Abductive Plan Recognition and Diagnosis: A Comprehensive Empirical Evaluation. In *KR*, 499–508.

Ovchinnikova, E.; Montazeri, N.; Alexandrov, T.; Hobbs, J. R.; McCord, M. C.; and Mulkar-Mehta, R. 2011. Abductive Reasoning with a Large Knowledge Base for Discourse Processing. In *IWCS*.

Poole, D. 1993a. Logic Programming, Abduction and Probability: a top-down anytime algorithm for estimating prior and posterior probabilities. *New Generation Computing* 11(3-4):377–400.

Poole, D. 1993b. Probabilistic Horn abduction and Bayesian networks. *Artificial Intelligence* 64 (1):81–129.

Poon, H., and Domingos, P. 2010. Unsupervised Ontology Induction from Text. In *ACL*, 296–305.

Raghavan, S., and Mooney, R. J. 2010. Bayesian Abductive Logic Programs. In *STARAI*, 82–87.

Santos, E. 1994. A linear constraint satisfaction approach to cost-based abduction. *Artificial Intelligence* 65 (1):1–27.

Sato, T. 1995. A statistical learning method for logic programs with distribution semantics. In *ICLP*, 715–729.

Schoenmackers, S.; Davis, J.; Etzioni, O.; and Weld, D. 2010. Learning First-order Horn Clauses from Web Text. In *EMNLP*, 1088–1098.

Suchanek, F. M.; Kasneci, G.; and Weikum, G. 2007. Yago: A Core of Semantic Knowledge. In *WWW*. ACM Press.