Vol.2011-NL-203 No.3
2011/9/16

An ILP Formulation of Abductive Inference for Discourse Interpretation

Naoya Inoue[†1] and Kentaro Inui[†1]

Abduction is inference to the best hypothesis to explain observations. Hobbs et al.[6] demonstrate that abduction gives a reasonable formalization of the process of discourse understanding, and several natural language processing (NLP) tasks can be resolved with a single abduction-based framework. However, there is a critical problem with this approach: the computational cost of abduction. The task of abductive reasoning quickly becomes intractable as the amount of background knowledge is increased to cover the millions of axioms necessary for robust discourse processing. This computational bottleneck is preventing abductive reasoning from benefiting from the recent advances in computational resources for for commonsense reasoning. In this paper, we propose an efficient implementation of Hobbs et al.'s abductive discourse interpretation framework, *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 an existing system for weighted abduction.

1. Introduction

Abduction is inference to the best hypothesis to explain observations using background knowledge. Abduction is widely used for such artificial intelligence systems as diagnostic systems that require finding an explanation to observations.

Applying abduction to NLP was pioneered by Hobbs et al.[6] They demonstrate that a wide range of tasks involved in discourse interpretation including anaphora resolution, discourse relation recognition can be cast as the problem of finding an explanation to the pieces of information observed from the discourse. Recent work in their group[12] conducts empirical evaluation of the framework through recognizing textual entailment (RTE) tasks, and reports that the performance is comparable to the state-of-the-art RTE systems. Plan recognition, the task of inferring an agent's plan from observed actions or utterances, is also a potential application to show the effectiveness of abductive discourse understanding[4),5),11].

The abduction-based approach to discourse understanding has several good advantages. It can inherently exploit background knowledge in the process of creating a plausible interpretation of a given discourse. It is also expected to provide a framework in which many types of linguistic processing can be formalized in an integrated fashion. In spite of those advantages, however, the models proposed in the 1980s and 1990s have not been tested on open data since they suffered from a shortage of background/world knowledge. In the several decades since, however, a number of methods for large-scale knowledge acquisition have been proposed[3),14),20),22], and the products of their efforts have been made available to the public. Now we are almost ready to test the effectiveness and robustness of abduction-based models with large-scale knowledge resources.

However, as the background knowledge is increased, the task of abductive reasoning quickly becomes intractable[2),12]. 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.'s *weighted abduction*[6], 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 describe the motivation of this work in more detail. We also discuss other existing implementation of abduction, including one of major probabilistic logic frameworks, Markov Logic Networks[18]. In Section 3, we describe the framework of weighted abduction, and then propose ILP formulation for weighted abduction in Section 4. 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.

†1 Graduate School of Information Sciences, Tohoku University

2. Background

In this section, we first give a brief review of logical abduction, and then elaborate on how abductive inference has been applied to framework of discourse interpretation and why we commit weighted abduction. Finally, we explain how our ILP-based framework relates to existing frameworks of abductive reasoning with uncertainty.

2.1 Abduction

Abduction is inference to the best explanation. Formally, logical abduction is defined as follows:

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

B is usually limited to a set of first-order Horn clauses, and both O and H are represented as a conjunction of ground positive literals. Typically, there are a number of hypotheses H that explains O. Abductive inference is to find the best hypothesis among competing hypotheses by a specific evaluation measure. We call each hypothesis H that explains O a *candidate hypothesis*, and a literal $h \in H$ as an *elemental hypothesis*. We call the best hypothesis H^* the *solution hypothesis*. The evaluation measures adopted by previous work range from the number of elemental hypotheses to the cost or probability of a hypothesis etc.

2.2 Motivation

Applying abduction to NLP was pioneered by Hobbs et al.[6] They demonstrate that a wide range of tasks involved in discourse interpretation including anaphora resolution, discourse relation recognition, etc. can be cast as the problem of finding an explanation to the pieces of information observed from the discourse. Although lack of computational resources of world knowledge in 1990s hampered empirical evaluation of this framework, recent work in his group[12] shows that the abduction-based framework is promising through empirical evaluation of Recognizing Textual Entailment (RTE) tasks on large dataset. It also reports that performance of Semantic Role Labeling (SRL) solved as a by-product of abductive inference is also comparable to systems designed specially for SRL. Abduction-based frameworks are the subject of study also in the field of plan recognition. Plan recognition is the task of inferring an agent's plan from observed actions or utterances, which is essential to tasks such as story understanding and dialogue planning. This task can also be naturally viewed as abductive reasoning, where an agent's plan is inferred by finding an explanation of observed actions or utterances. A number of papers have been published in this line of study[4,5,8,11,16,21].

Hobbs et al. extended the framework of abductive discourse interpretation to evaluate both specificity and likelihood of hypothesis in best explanation finding. This extended framework is called *weighted abduction*. It gives a cost to each explanation as the evaluation measure to choose the best interpretation which is an appropriate level of specificity and occurs most likely, as detailed in Section 3. To the best of our knowledge, weighted abduction is the only framework that has the mechanism of quantifying the appropriate level of hypothesis specificity. In abduction-based discourse processing, it is crucial to discuss what level of explanation specificity would be appropriate for abductive reasoning system. 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. Most of plan recognition work falls into this group[5,8,16]. The second is *least-specific abduction*. Literally, in this mode an explanation is just assuming observations. We need this mode in some cases of natural language understanding. Adopting only one of these levels is problematic. For example, suppose we want to recognize character's intention from his action: "*Bob took a gun to a bank.*" If we adopt most-specific abduction, the plan recognition system yields too specific explanation such as "*Bob took a gun to a bank because he would rob XYZ bank using a machine gun which he had bought three days ago.*" We want to avoid hypothesizing too specific information such as when he bought a gun or which bank he went to, since the observation gives little evidence to the determination of such information. Conversely, if we adopt least-specific abduction, the system assumes just observation, as in "*Bob took a gun to a bank.*" Although uncertain explanation as described above should not be inferred, it is useless if all of the observations are just assumed. We thus want to determine the suitable specificity during inference so that more evidential hypothesis would be inferred, and less evidential hypothesis would not be inferred. Therefore, we adopt Hobbs et al[6]'s weighted abduction in order to represent the specificity of hypotheses.

One issue of concern about weighted abduction is semantics of cost and weight. It has long been pointed out that the semantics is unclear. However, we would overcome this weakness by giving probabilistic interpretation such as Blythe et al.'s semantics[2]⋆1.

2.3 Related work

A number of frameworks that integrate logical inference and uncertainty have been proposed in recent years. One of the major frameworks is Markov Logic Networks[18] (MLNs). There are several studies for performing probabilistic abduction in MLNs[2),8),21]. For instance, Blythe et al.[2] demonstrate that weighted abduction can be mapped into weighted logical formulae in MLNs. MLN-based approaches, however, require special procedures to convert abduction problem into deduction problem because of the deductive nature of MLNs: an MLN which consists of background axioms in abduction cannot be used for traditional probabilistic inference for abduction, MAP estimation given observations. Thus, the pioneering work of MLN-based abduction[8] converts background axioms into MLN logical formulae by (i) reversing implication and (ii) constructing axioms representing mutual exclusiveness of explanation (e.g., the set of background knowledge axioms $\{p_1 \Rightarrow q, p_2 \Rightarrow q, p_3 \Rightarrow q\}$ is converted into the following MLN formulae: $\{q \Rightarrow p_1 \vee p_2 \vee p_3,\ q \Rightarrow \neg p_1 \vee \neg p_2,\ q \Rightarrow \neg p_1 \vee \neg p_3,\ q \Rightarrow \neg p_2 \vee \neg p_3\}$). As you can imagine, MLN-based approach suffers from the slow inference speed due to the increase of converted axioms; thus Singla and Mooney[21] propose a method to reduce the converted axioms.

However, we don't see any particular reason to adopt the deduction-based framework to perform abductive inference. One would think that MLN-based approach has a several good points to perform probabilistic logical inference: it has a framework of parameter learning in a supervised fashion and full First-Order Logic (FOL) expressiveness. However, learning framework of weighted abduction weights can be independently developed, whatever inference framework we use. We would think several possible ways of parameter learning: a probabilistic learning based on semantics of cost and weight mentioned in Section 2.2, perceptron-style learning etc. Regarding expressiveness, our framework also accepts almost full FOL in natural way as described in Section 4. Therefore, our ILP-based framework would be a promising alternative to MLN-based approaches.

There are also other choices to implement abductive reasoning with uncertainty[7),13),15),19]. One of the major frameworks followed by many other researches is Santos Jr.'s ILP implementation[19] of cost-based abduction[5]. He formalized cost-based abduction as a linear constraint satisfaction problem, and efficiently obtained the best hypothesis by solving the problem with the ILP technique. He converted propositions generated during abductive inference into ILP variables, and used the sum-product of these variables and the costs as the ILP objective function. Our approach also adopts ILP formulation, and performs a similar translation. However, his approach is based on propositional logic, and assumes that set of assumable literals is given: it is incapable of evaluating appropriate level of hypothesis specificity. The comparison with our approach is more detailed in Section 4. For small-scale reasoning, Mulkar et al.'s Mini-TACITUS[9] would be a possible choice, however, it often fails to give an optimal solution hypothesis to large dataset in practical time[12] as shown in Section 5.

3. Weighted abduction

Hobbs et al.[6] 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 ab-*

⋆1 As far as we investigated, their interpretation is partially incomplete. However, we make sure that the interpretation can be plausible by making small modification. We will not discuss the semantics here since it is beyond the scope of this paper.

duction. 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 6), 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 positive 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 positive 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 positive 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:

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

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 Formulation of weighted abduction

Now we describe our ILP-based formulation of weighted abduction. Our framework accepts the following format of input and output:

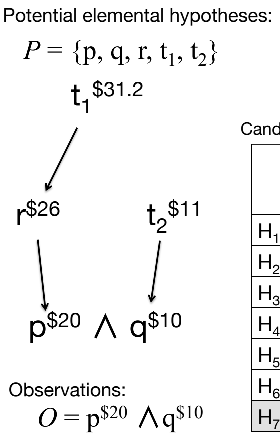
- a set of logical formulae[⋆1] as background knowledge, and;
- an existentially quantified conjunction of literals or inequalities of existentially quantified variables as observations, and;
- an existentially quantified conjunction of literals as a solution hypothesis.

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 hypothesis generation

Let B be background knowledge, O be observations and $\mathcal{H} = \{H_1, H_2, ...H_n\}$

⋆1 Non-Horn clauses in predicate logic, such as $p(x) \wedge q(z) \Rightarrow r(x) \wedge s(y, z) \wedge t(z)$), cannot not be processed correctly in the current formulation. However, there are several strategies for handling this type of clause in a natural way.



Background knowledge:

$B = \{r^{1.3} \Rightarrow p,\ t^{1.1} \Rightarrow q,\ t^{1.2} \Rightarrow r\}$

Candidate hypotheses:

$\mathcal{H}$		p	q	r	t_1	t_2	$c(H)$	
H_1	$p \wedge q$	●	●				\$30	
H_2	$p \wedge t_2$	●	○			●	\$31	
H_3	$r \wedge q$	○	●	●			\$36	
H_4	$r \wedge t_2$	○	○	●		●	\$37	
H_5	$t_1 \wedge q$	○	●	○	●		\$41.2	
H_6	$t_1 \wedge t_2$	○	○	○	●	●	\$42.2	
H_7	t_2	○	○	○	○	●	\$11	← H^*

* t_1 and t_2 are unified in H_7.

Fig. 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 $H_i \cup B$.

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*).

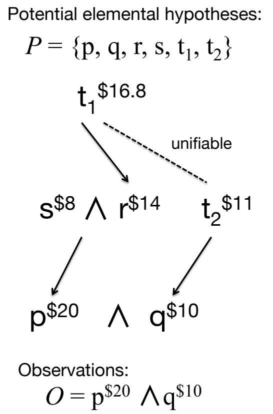
Initialization

$$H \leftarrow O \tag{4}$$

Backward reasoning

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

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



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: $r_p \le h_s + h_r,\ h_s = h_r \cdot 1$

$r_{t1} \le u_{t1,t2}$

C3: $u_{t1,t2} \le \frac{1}{2}(h_{t1} + h_{t2})$

Candidate hypotheses:

$\mathcal{H}$		h_p	r_p	h_q	r_q	h_r	r_r	h_s	r_s	h_{t1}	r_{t1}	h_{t2}	r_{t2}	$u_{t1,t2}$	$c(H)$
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

Fig. 2 ILP representation for the space of candidate hypotheses in the case for propositional logic

Unification

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

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

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 is intractable.

4.2 ILP 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:

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

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:

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

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:

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

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 $p_1, p_2 \in P$ can be unified (i.e., $u_{p_1,p_2} = 1$) only if both p_1 and p_2 are included in or explained by a candidate hypothesis (i.e., $h_{p_1} = 1$ and $h_{p_2} = 1$).

$$u_{p_1,p_2} \leq \frac{1}{2}(h_{p_1} + h_{p_2}) \text{ for each } p_1, p_2 \in P \tag{13}$$

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.

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 ILP variables. Hereafter, we use V to denote a set of existentially quantified variables in P, C to denote a set of non-skolem constants in P[⋆1], U to denote a set of pairs of unifiable literals in P, and θ_U to denote all possible variable substitutions to take place in U. We describe how to handle variable substitution by using Figure 3.

Our framework recasts the notion of variable substitution as clustering of each element in $V \cup C$ to equivalent group: it has set E of clusters of equivalent class (hereafter, *equivalent cluster*) such that variables and constants which are assumed equal would belong to (e.g., if $x = y, y = z$ are assumed, x, y, z are in the same cluster)[⋆2]. For example, in Figure 3, we have three equivalent clusters, and $x_1 = x_2$ is assumed. We first define new ILP variables to express the cluster assignment of variable and constant, and then introduce constraints which are imposed on unification variables.

Cluster assignment Each variable or constant $x \in V \cup C$ can be assigned to the single equivalent cluster $t \in E$. We introduce the new variable $c \in \{0, 1\}$ defined as follows:

$$c_{x,t} = \begin{cases} 1 & \text{if } x \text{ is assigned to the equivalent cluster } t \\ 0 & \text{otherwise} \end{cases} \quad \text{for each } x \in V \cup C$$

In Figure 3, $c_{x_1,1}$ and $c_{x_2,1}$ are set to 1 since the logical variables x_1, x_2 are in the equivalent cluster 1.

In order to guarantee that (i) each variable or constant is assigned to at most one cluster, and (ii) variable substitution is logically consistent (i.e., two distinct constants must be assigned to different clusters), we also introduce two basic constraints on c as follows.

Constraint 4 Each variable or constant x can be assigned to at most single cluster (i.e., $c_{x,t} = 1$ can be held for at most one $t \in E$). This constraint is expressed as:

$$\sum_{t \in E} c_{x,t} \leq 1 \text{ for each } x \in V \cup C \tag{14}$$

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

⋆2 We adopt the notion of clustering instead of pairwise variable substitution x/y, because we can avoid writing ${}_nC_3 \times 3$ ILP constraints for transitivity relation of equality between elements in $V \cup C$, where $n = |V \cup C|$.

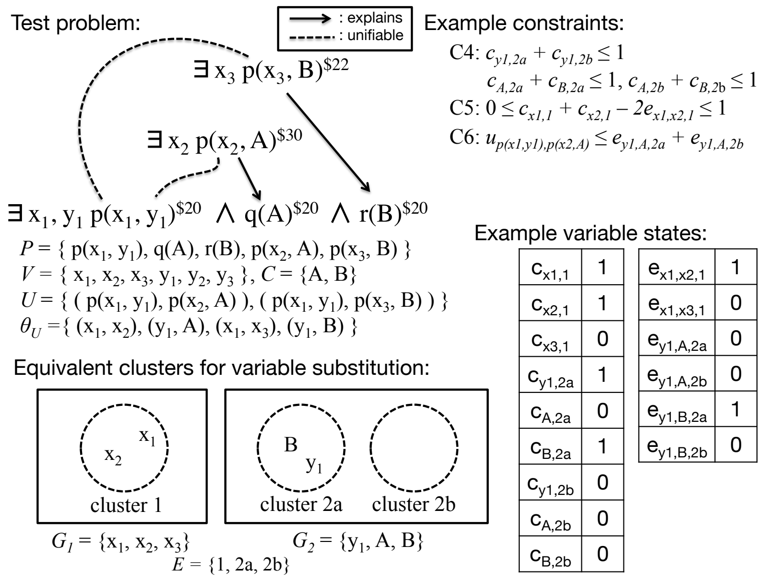


Fig. 3 Unification in ILP framework for the case of first-order logic

In Figure 3, $c_{y_1,2a}+c_{y_1,2b} \leq 1$ is created because y_1 can be assigned to at most one cluster among $\{2a, 2b\}$. Note that the cluster 1 need not be involved here since y_1 would not be clustered to 1 as described later. The next constraint we introduce is about clustering of constants: assuming that different constants are equal is not allowed. Formally, the constants $a_1, a_2, ..., a_n \in C$ must not be assigned to the same cluster t (i.e., either $c_{a_1,t}$, $c_{a_2,t}$, ..., or $c_{a_n,t}$ can be 1). This constraint can be expressed as the following inequality:

$$\sum_{a \in C} c_{a,t} \leq 1 \quad \text{for each} \quad t \in E \tag{15}$$

In Figure 3, $c_{A,2a} + c_{B,2a} \leq 1$ and $c_{A,2b} + c_{B,2b} \leq 1$ are generated since the constants A and B cannot be assumed equal.

Given set P of potential elemental hypotheses, we prepare ILP variable c as the following procedure. First of all, we find out θ_U, i.e., what variable substitution can potentially occur by scanning U. For instance, in Figure 3, we can see that $x_1 = x_2$ can possibly occur etc. The second step is to group them by equivalent class, which yields N equivalent sets G_i of variable or constants. Here we get $N = 2$ sets: $G_1 = \{x_1, x_2, x_3\}$ and $G_2 = \{y_1, A, B\}$. In the third step, we prepare N equivalent clusters in E, and divide each cluster into the number of constants included in each group since constants need to be in separate clusters. In Figure 3, the second cluster is divided into $2a$ and $2b$ since there are two constants A and B in the group. Finally, we assign ILP variable c to each pair of $x \in V \cup C$ and $t \in E$. Note that we don't need to consider any combination of $V \cup C$ and E, because most of substitution in θ_U would be expressed as its partial combinations. For instance, in Figure 3, it is not necessary to generate $c_{y_1,1}$ but only $c_{y_1,2a}$ and $c_{y_1,2b}$ since y_1 will not be substituted with the elements of G_1.

As mentioned above, we also use additional constraints imposing on unification so that the framework checks the states of variable substitutions needed for the unification. One idea of writing unification constraint is: two literals $p(x), p(y) \in U$ can be unified (i.e., $u_{p(x),p(y)} = 1$) only if both x and y are cluster 1 (i.e., $c_{x,1} = 1 \wedge c_{y,1} = 1$), both x and y are cluster 2 (i.e., $c_{x,2} = 1 \wedge c_{y,2} = 1$), etc.; however, in order to write ILP constraints based on the idea, we would have to generate $R \cdot 2^T$ ILP constraints⋆1 for each pair of unifiable literals, where R is the number of arguments in unified literals and T is the number of cluster. In order to suppress this, we implement unification constraint by using another ILP variable e expressing $c_{x,t} = 1 \wedge c_{y,t} = 1$ instead of c⋆2. We first introduce the intermediate ILP variable e, and then define constraint that retains relation between c and e.

Pairwise equality We introduce the new variable $e \in \{0, 1\}$ defined as:

$$e_{x,y,t} = \begin{cases} 1 & \text{if } x \text{ and } y \text{ are assigned to the cluster } t \\ 0 & \text{otherwise} \end{cases} \quad \text{for each} \ \ (x, y) \in \theta_U$$

In Figure 3, $e_{x_1,x_2,1}$ is generated since x_1 and x_2 are possible variable substitution for $p(x_1, y_1)$ and $p(x_2, A)$. The value is set to 1 because x_1 and x_2 are in the cluster 1.

⋆1 The logical form of this idea is represented as $u_{p(x),p(y)} = 1 \Rightarrow (c_{x,1} = 1 \wedge c_{y,1} = 1) \vee (c_{x,2} = 1 \wedge c_{y,2} = 1) \vee \ldots$. We need a conjunctive normal form (CNF) to generate ILP constraints, and generate one ILP constraint for each conjunct[17]. In this case, the transformed CNF would consist of 2^T conjuncts.

⋆2 By using e, the former idea can be represented as: $u_{p(x),p(y)} = 1 \Rightarrow e_{x,y,1} = 1 \vee e_{x,y,2} = 1 \vee \ldots$. Since it is already CNF, it can be expressed as a single ILP constraint, not 2^T.

Constraint 5 $e_{x,y,t}$ is set to 1 if and only if x and y belong to the same equivalent cluster t (i.e., $\exists t(c_{x,t}=1 \wedge c_{y,t}=1)$). To express this consistency, we introduce the following inequality:

$$0 \le c_{x,t}+c_{y,t}-2\cdot e_{x,y,t} \le 1 \text{ for each } (x,y)\in\theta_U, t\in E \quad (16)$$

In Figure 3, $0 \le c_{x_1,1}+c_{x_2,1}-2\cdot e_{x_1,x_2,1} \le 1$ is introduced to express both x_1 and x_2 are assigned to the cluster 1.

We are now ready to introduce constraint imposed on unification of literals.

Constraint 6 Two unifiable literals $(p_1(x_1,x_2,...,x_n), p_2(y_1,y_2,...,y_n)) \in U$ are allowed to be unified (i.e., $u_{p_1(x_{1:n}),p_2(y_{1:n})}=1$)*1 only when all pairs of i-th arguments of p_1 and p_2 are assumed equal (i.e., $\forall i \exists t(e_{x_i,y_i,t}=1)$).

$$u_{p_1(x_{1:n}),p_2(y_{1:n})} \le \sum_{t\in E} e_{x_i,y_i,t} \text{ for each } (x_i,y_i)\in\theta, \quad (17)$$

where θ denotes a set of pairs of i-th arguments of p_1 and p_2. In Figure 3, the constraint $u_{p(x_1,y_1),p(x_2,A)} \le e_{y_1,A,2a}+e_{y_1,A,2b}$ is generated since y_1 needs to be substituted for A when $p(x_1,y_1)$ and $p(x_2,A)$ are unified. Furthermore, in order to prohibit that variables or constants are assumed equal without unification of literals, we introduce the additional constraint: variables or constants $(x,y)\in\theta_U$ are allowed to be assumed equal (i.e., $\exists t(e_{x,y,t}=1)$) only if at least one pair of unifiable literals $(p_1,p_2)\in U$ which have x,y as their i-th argument are unified.

$$\sum_{t\in E} e_{x,y,t} \le \sum_{(p_1,p_2)\in uni(x,y)} u_{p_1,p_2} \text{ for each } (x,y)\in\theta_U, \quad (18)$$

where $uni(x,y)$ denotes set of pairs of unifiable literals requiring variable substitution x,y for the unification*2.

Our approach is different from Santos[19]'s LP formulation in terms that our approach is capable of evaluating the specificity of hypotheses, as mentioned in Section 2.3. 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. Furthermore, our framework can handle negation while Santos Jr.'s framework assumes positive literals as its input.

Finally, let us describe how inequality of variables and negation of literal are implemented in our framework. The inequality is simply expressed through constraint on ILP variable c similar to Constraint 4 to express mutual exclusiveness of clustering of constants.

Constraint 7 Two variables $x,y\in V$ are not allowed to be assumed equal (i.e., $c_{x,t}$ and $c_{y,t}$ cannot be 1 simultaneously for each t) if $x\ne y$ appears in observations.

$$c_{x,t}+c_{y,t}\le 1 \text{ for each } x\ne y\in O, t\in E \quad (19)$$

In addition to this constraint, we need new procedure of preparing set E of equivalent clusters described above. It is because x and y must belong to different clusters in E. The new procedure is similar to the separation of clusters for constants: when we prepare E, we divide equivalent clusters in the number of variables participating in inequalities in G_i. For instance, suppose we have $x_1\ne x_2$ in observations. Then we would divide cluster 1 into $1a$ and $1b$ in the preparation step so that x_1 and x_2 can be assigned to different clusters.

We then introduce how to handle negation of literal.

Constraint 8 For any combination of $p(x)\in P$ and $Q\in\{Q \mid Q\subseteq P, \exists\theta(Q\theta\cup B\models\neg p(x))\}$, a contradiction arises only if (i) $p(x)$ and all the literals in Q are hypothesized, and (ii) all the substitutions in θ take place. To avoid the contradiction, we impose: all the literals in Q are allowed to be hypothesized simultaneously (i.e., $h_{p(x)}+\sum_{q(y)\in Q} h_{q(y)}=1+|Q|$) can be held) only if θ does *not* take place (i.e., $\sum_{(x,y)\in\theta}\sum_{t\in E} e_{x,y,t}\le|\theta|-1$). This can be expressed as the following inequality:

---

*1 $x_{1:n}$ is a short notation for $x_1,x_2,...,x_n$.

*2 Actually, formula (17), (18) can be reduced to the following inequality:

$$|uni(x,y)|+1 \le \sum_{(q_1,q_2)\in uni(x,y)} u_{q_1,q_2} - |uni(x,y)|\cdot\sum_{t\in E} e_{x,y,t} \le 0 \text{ for each } (x,y)\in\theta_U$$

$$h_{p(x)} + \sum_{q(y)\in Q} h_{q(y)} - |Q| \le |\theta| - \sum_{(x,y)\in\theta} \sum_{t\in E} e_{x,y,t} \quad (20)$$

Note that if Q leads to a contradiction without variable substitution (i.e., $\theta = \{\}$), inequality (20) is reduced to the following simple inequality:

$$h_{p(x)} + \sum_{q(y)\in Q} h_{q(y)} \le |Q| \quad (21)$$

However, it is still not clear how to control the overall process of searching the set Q of literals that can cause a contradiction. This search requires us to consider complicated cases; namely we need to implement forward reasoning (e.g., consider the axiom $\forall x(male(x) \Rightarrow \neg female(x))$). Its computational difficulty comes from nature of logical inference. One possible and plausible heuristic is that we consider first-order forward reasoning: searching (i) $\neg p(y) \in P$ such that $\exists\theta(\neg p(y)\theta \models \neg p(x))$, and (ii) $Q \subseteq P$ such that $\exists\theta(Q\theta \cup B_1 \models \neg p(x))$, where B_1 is set of background axioms that contain $\neg p(x)$ as their consequent.

5. Evaluation

We evaluated the efficiency of our ILP-based framework on two datasets 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 the parameter d of experiment setting, 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[⋆1][19], which is the state-of-the-art tool of weighted abduction. To the best of our knowledge, Mini-TACITUS is the only tool of weighted abduction available now. We have investigated (i) how many problems in our testset Mini-TACITUS could solve in 600 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[⋆2], which is the fastest solver among non-commercial solvers. SCIP solves ILP problems using the *branch-cut-and-price* method.

5.1 Dataset

Story Understanding Our first test set was extracted from the dataset originally developed for Ng and Mooney[11]'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 a conjunction of literals. For example, in the problem *t2*, the following observation literals are provided:

(1) $inst(get2, getting) \wedge agent\_get(get2, bob2) \wedge name(bob2, bob) \wedge patient\_get(get2, gun2) \wedge inst(gun2, gun) \wedge ...$

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.

Monroe Since story understanding dataset is fairly small, we explore the scalability of our framework on a larger-scale dataset. We thus extracted plan

⋆1 http://www.rutumulkar.com/

⋆2 http://scip.zib.de/

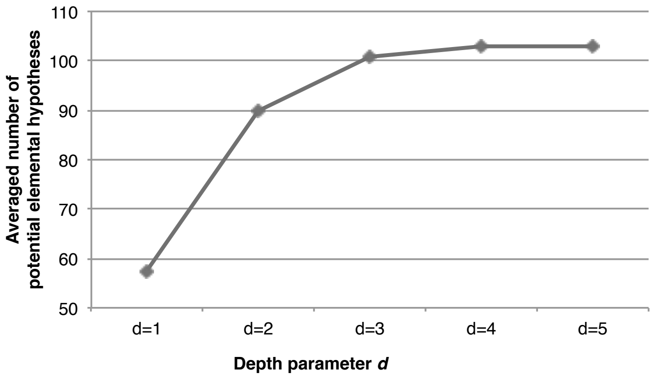


Fig. 4 Complexity of each problem setting

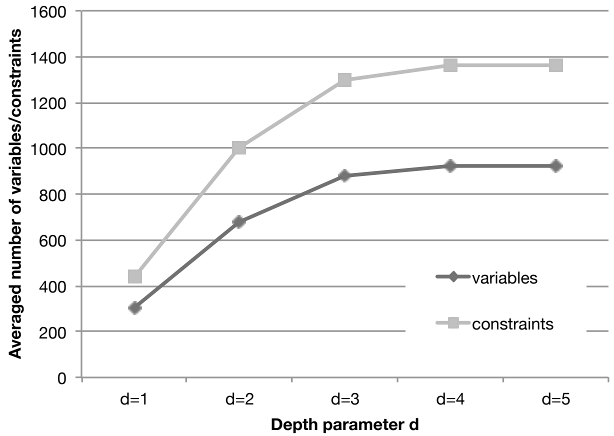


Fig. 5 Complexity of each ILP problem

recognition problems from Blaylock and Allen's Monroe Plan Corpus[1]. This corpus includes 5,000 plans, each of which consists of hierarchical goal-subgoal relations artificially generated by SHOP2 planning system[10]. We extracted 202 background axioms, and used lowest-level actions of 500 plans as input of our system. We assigned the weights of axioms equally so that the sum of the weights is 1.2.

5.2 Results and discussion

First of all, let us show the complexity of abduction problems in *testset A*. Figure 4 shows the number of potential elemental hypotheses, P described in Section 4, averaged for all the problems. Recall that the number of candidate hypotheses is $O(2^n)$, where n is the number of potential elemental hypotheses ($|P|$). Therefore, in *testset A*, we roughly have $2^{100} \approx 1.3 \cdot 10^{30}$ 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 all 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 non-exponential time to the size of the candidate hypotheses which increases $O(2^{|P|})$.

Table 1 shows the inference time of ILP-based framework and complexity of solved problems on Monroe Dataset. Again, the inference time indicates that our framework solves abduction problems efficiently in the setting $d = 1, d = 2$. In $d = 3$ setting, however, the inference time unexpectedly blows up. Our framework could not solve 3.2% (16/500) of the problems in 600 seconds, and it took 17.4 seconds on average.

Our analysis revealed that the number of variables u, c, e and constraints imposed on these variables was very large. The reason was that Monroe Plan Corpus had a couple of abstract actions entailed by a number of subgoals (e.g., *get-to*/2 is entailed by 24 subgoals), and many background axioms that have existentially quantified variables. To avoid the undesirable growth of unifiable literals, we could adopt heuristics to filter out unnecessary unification before we apply abductive reasoning to a discourse. Since our goal is to develop the framework of discourse understanding, we could definitely utilize some properties of discourse for such filtering. For instance, we could use linguistic knowledge such

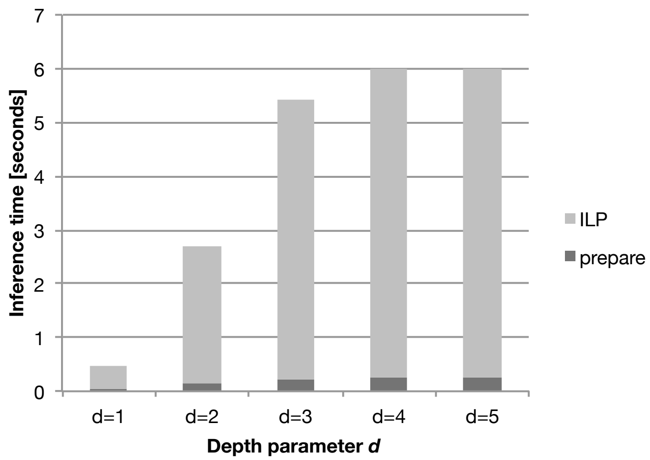


Fig. 6 Results of ILP-based inference time on Story Understanding dataset
"prepare" and "ILP" denote the time required to convert a weighted abduction problem to ILP problem, and the time required to solve the ILP problem respectively.

as the saliency of discourse entities. We could retain top-k entities based on its saliency, and allow only them to be substituted.

Then we show the inference time of Mini-TACITUS on *testset B* and Monroe Dataset. The complexity of *testset B* 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 5.5 seconds when averaged for the 50 problems of $d = 5$. The results of abductive reasoning on Mini-TACITUS is shown in Table 2. In Story Understanding dataset, the results show that the 58.0% of the problems (29/50) could not be solved in 600 seconds for the easiest setting $d = 1$. For the slightly complex setting $d \geq 2$, 72.0% of the problems (36/50) could not be solved in 600 seconds. We found that no additional axioms were applied in the 14 solved problems for $d \geq 3$: the search space did not change. In Monroe Dataset, the results also show that Mini-TACITUS could not solve

Table 1 Results of ILP-based framework on Monroe Dataset

	% of solved	Avg. of inference times [sec.]	P	V	C
d=1	100.0% (500/500)	0.02 (0.01/0.01)	19.3	44.8	35.5
d=2	100.0% (500/500)	0.03 (0.01/0.02)	29.2	66.2	68.2
d=3	96.8% (484/500)	17.4 (0.1/17.3)	176.5	1289.6	1662.9
d=4	75.2% (376/500)	59.2 (1.9/57.3)	265.4	2389.0	2935.4

"% of solved" indicates ratio of problems each system could solve in 600 seconds to all the problems. "Avg. of inference times" indicates inference time averaged over solved problems, with the time required to convert a weighted abduction problem to ILP problem, and the time required to solve the ILP problem. P, V, C indicates the averaged number of potential elemental hypotheses, ILP variables, and ILP constraints respectively.

Table 2 Results of weighted abduction on Mini-TACITUS

Dataset		% of solved	Avg. of inference times [sec.]
Story	$d = 1$	42.0% (21/50)	63.7
	$d = 2$	28.0% (14/50)	30.3
	$d = 3$	28.0% (14/50)	30.3
Monroe	$d = 1$	65.6% (328/500)	34.2
	$d = 2$	32.2% (161/500)	34.6
	$d = 3$	32.0% (160/500)	33.8

"% of solved" indicates that the ratio of problems Mini-TACITUS could solve in 600 seconds to all the 50 problems. "Avg. of inference times" denotes the inference time averaged for the solved problems.

34.4% of the problems (172/500) for the easiest setting $d = 1$ and 67.8% of the problems (339/500) in the slightly complex setting $d = 2$. 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. Conclusion

Weighted abduction is promising as a universal framework for representing linguistic information necessary for discourse understanding. However, discourse understanding requires large amounts of world knowledge. As large-scale linguistic resources have been developed and released to the public, there has been a resurgence in research on abductive logic. Given these background, we have addressed the scalability issue of abductive reasoning, and proposed an ILP-based framework for weighted abduction, which maps abductive inference problem to a linear programming problem and efficiently finds an optimal solution. Our qualitative comparison to other existing work showed that our framework can be a good alternative to other work. The results of our experiments have demonstrated that our approach efficiently solved the problems of abduction, and it is an encouraging level of performance for investigating the scalability of our framework to real world problems. Our future work includes testing on larger discourse understanding dataset as used in 12), and exploring automatic assignment of axiom weights. We also plan to explore how search heuristics such as contradiction search mentioned in Section 4 and unification candidate reduction mentioned in Section 5.2 can be incorporated into our framework.

Acknowledgments

The authors would like to thank Eric Nichols for valuable comments. This work was supported by Grant-in-Aid for JSPS Fellows (22-9719) and KAKENHI (23240018).

References

1) Blaylock, N. and Allen, J.: Generating Artificial Corpora for Plan Recognition, *UM*, Springer (2005).
2) Blythe, J., Hobbs, J.R., Domingos, P., Kate, R.J. and Mooney, R.J.: Implementing Weighted Abduction in Markov Logic, *IWCS* (2011).
3) Chambers, N. and Jurafsky, D.: Unsupervised Learning of Narrative Schemas and their Participants, *ACL*, pp.602–610 (2009).
4) Charniak, E. and Goldman, R. P.: A Probabilistic Model of Plan Recognition, *AAAI*, pp.160–165 (1991).
5) Charniak, E. and Shimony, S. E.: Cost-based abduction and map explanation, *Artificial Intelligence*, Vol.66(2), pp.345–374 (1994).
6) Hobbs, J.R., Stickel, M., Appelt, D. and Martin, P.: Interpretation as Abduction, *Artificial Intelligence*, Vol.63, pp.69–142 (1993).
7) Ishizuka, M. and Matsuo, Y.: SL Method for Computing a Near-optimal Solution using Linear and Non-linear Programming in Cost-based Hypothetical Reasoning, *PRCAI*, pp.611–625 (1998).
8) Kate, R.J. and Mooney, R.J.: Probabilistic Abduction using Markov Logic Networks, *PAIRS* (2009).
9) Mulkar, R., Hobbs, J.R. and Hovy, E.: Learning from Reading Syntactically Complex Biology Texts, *The 8th International Symposium on Logical Formalizations of Commonsense Reasoning* (2007).
10) Nau, D., Au, T., Ilghami, O., Kuter, U., Murdock, J.W., Wu, D. and Yaman, F.: SHOP2: An HTN planning system, *Journal of Artificial Intelligence Research*, Vol.20, pp.379–404 (2003).
11) Ng, H.T. and Mooney, R.J.: Abductive Plan Recognition and Diagnosis: A Comprehensive Empirical Evaluation, *KR*, pp.499–508 (1992).
12) Ovchinnikova, E., Montazeri, N., Alexandrov, T., Hobbs, J. R., McCord, M. C. and Mulkar-Mehta, R.: Abductive Reasoning with a Large Knowledge Base for Discourse Processing, *IWCS* (2011).
13) Poole, D.: Logic Programming, Abduction and Probability: a top-down anytime algorithm for estimating prior and posterior probabilities, *New Generation Computing*, Vol.11(3-4), pp.377–400 (1993).
14) Poon, H. and Domingos, P.: Unsupervised Ontology Induction from Text, *ACL*, pp.296–305 (2010).
15) Prendinger, H. and Ishizuka, M.: First-order diagnosis by propositional reasoning: A representation-based approach, *DX*, pp.220–225 (1999).
16) Raghavan, S. and Mooney, R.J.: Bayesian Abductive Logic Programs, *STARAI*, pp.82–87 (2010).
17) Raman, R. and Grossmann, I.E.: Relation between MILP modelling and logical inference for chemical process synthesis, *Computers & Chemical Engineering*, Vol.15 (2), pp.73–84 (1991).
18) Richardson, M. and Domingos, P.: Markov logic networks, *Machine Learning*, pp. 107–136 (2006).
19) Santos, E.: A linear constraint satisfaction approach to cost-based abduction, *Artificial Intelligence*, Vol.65 (1), pp.1–27 (1994).
20) Schoenmackers, S., Davis, J., Etzioni, O. and Weld, D.: Learning First-order Horn Clauses from Web Text, *EMNLP*, pp.1088–1098 (2010).
21) Singla, P. and Mooney, R.J.: Abductive Markov Logic for Plan Recognition, *AAAI*, pp.1069–1075 (2011).
22) Suchanek, F.M., Kasneci, G. and Weikum, G.: Yago: A Core of Semantic Knowledge, *WWW*, ACM Press (2007).