Large-scale Cost-based Abduction in Full-fledged First-order Predicate Logic with Cutting Plane Inference

Naoya Inoue and Kentaro Inui

Tohoku University, 6-3-09 Aramaki Aza Aoba, Aobaku, Sendai 980-8579, Japan
{naoya-i,inui}@ecei.tohoku.ac.jp

Abstract. Abduction is inference to the best explanation. Abduction has long been studied intensively in a wide range of contexts, from artificial intelligence research to cognitive science. While recent advances in large-scale knowledge acquisition warrant applying abduction with large knowledge bases to real-life problems, as of yet no existing approach to abduction has achieved both the efficiency and formal expressiveness necessary to be a practical solution for large-scale reasoning on real-life problems. The contributions of our work are the following: (i) we reformulate abduction as an Integer Linear Programming (ILP) optimization problem, providing full support for first-order predicate logic (FOPL); (ii) we employ Cutting Plane Inference, which is an iterative optimization strategy developed in Operations Research for making abductive reasoning in full-fledged FOPL tractable, showing its efficiency on a real-life dataset; (iii) the abductive inference engine presented in this paper is made publicly available.

Keywords: abduction, cost-based abduction, cutting plane inference, integer linear programming

1 Introduction

Abduction is inference to the best explanation. Abduction has long been studied in a wide range of contexts. For example, abduction has been viewed as a promising framework for describing the mechanism of human perception [1–4, etc.]. The idea is that the declarative nature of abduction enables us to infer the most plausible, implicitly stated information combining several types of inference, and pieces of explicitly observed information, as humans do. Hobbs et al. [2] showed the process of natural language interpretation can reasonably be described as abductive inference; finding the lowest-cost abductive proof provides the solutions to a broad range of natural language pragmatics problems, such as word sense disambiguation, anaphora, and metonymy resolution. It will be a significant contribution for such research areas if we could provide an efficient abductive reasoning engine which scales to large problems.

In this paper, we explore first-order predicate logic-based abduction *with large knowledge bases (KBs) for solving "real-life" problems.* While the lack of world knowledge resources hampered applying abduction to real-life problems in the 1980s and 1990s, a number of techniques that acquire world knowledge resources have been developed in the last decade [5–9, etc.]. Consequently, several researchers start applying abduction to real-life problems, exploiting large KBs. For instance, inspired by Hobbs et al. [2], Ovchinnikova et al. [10] propose an abduction-based natural language processing framework using forty thousands of axioms extracted from the popular ontological resources, WordNet [5] and FrameNet [6]. They evaluate their approach on the real-life natural language processing task of Recognizing Textual Entailment (RTE) [11].

However, in order to apply large-scale abductive inference to real-life problems, we still need to address the following issue: *how to search for the best explanation efficiently.* Abduction is known to be an NP-hard problem in general [12]; this hampers the application of abduction with large knowledge resources to real-life problems. In fact, Ovchinnikova et al. [10] report that the Mini-TACITUS abductive reasoning system [13] could not search the entire search space of explanations within 30 minutes in most of the RTE problems in their experiments. In the literature, many researchers have tried to overcome abduction's inefficiency by a range of methods from approximation [14–16, etc.] to exact inference [17, 18, etc.]. However, most of the proposed methods are optimized for propositional logic in principle. Inoue and Inui [18] provides an efficient approach to first-order predicate logic (FOPL)-based abduction, showing superior efficiency to Mini-TACITUS system [13]; however, it does not provide full support of FOPL (e.g. negation is not supported). In addition, as the reader will see in Sec. 3.2 and Sec. 4, it does not scale to larger problems due to the intractability emerging from the use of FOPL inference.

In this paper, we address this issue with the following contributions:

(i) we extend Inoue and Inui [18]'s Integer Linear Programming (ILP)-based inference method, *providing full support for FOPL including negation*;
(ii) we describe how Cutting Plane Inference, an iterative optimization strategy developed in Operations Research, can be exploited for *making abductive reasoning in full-fledged FOPL tractable*, showing its efficiency by *providing evaluation on a large, real-life dataset*;
(iii) the abductive inference engine presented in this paper is made publicly available.

The structure of our paper is as follows. We start with a brief introduction of *cost-based* abduction, where the quality of explanation is evaluated by some cost function (Sec. 2.1). We then briefly describe Inoue and Inui [18]'s ILP-based formulation of cost-based abduction (Sec. 2.2). In the next section, we first describe how their formalization can be extended for handling negation (Sec. 3.1). We then show how Cutting Plane Inference (CPI) enables us to apply abductive reasoning in full-fledged FOPL with large KBs (Sec. 3.2). Finally, we evaluate the efficiency of our CPI-based framework on a large, real-life problem of natural language processing (Sec. 4), and give a comparison of our work with the prior implementations of cost-based abduction (Sec. 5).

2 Background

2.1 Cost-based abduction

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

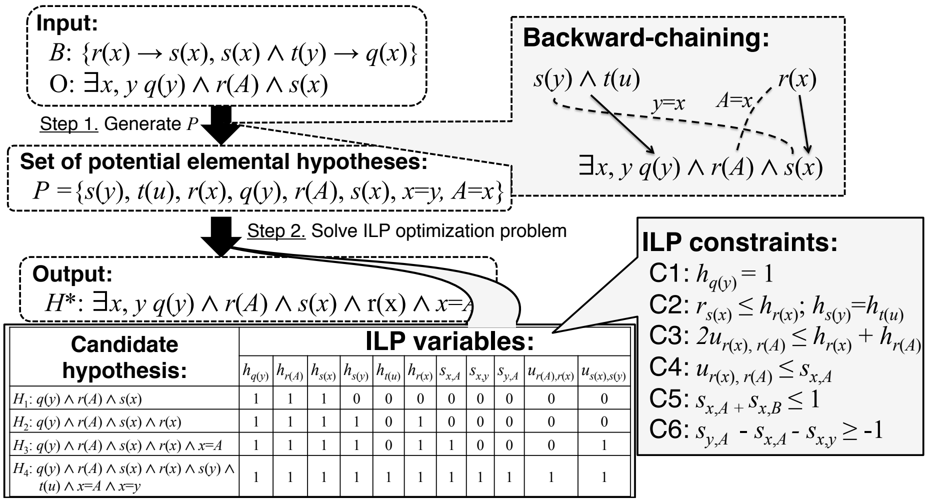
- **Given:** Background knowledge B, and observations O, where both B and O are sets of first-order logical formulas
- **Find:** A *hypothesis* (or *explanation*) H such that $H \cup B \models O, H \cup B \not\models \perp$, where H is a set of first-order logical formulas. We say that p is *hypothesized* if $H \cup B \models p$, and that p is *explained* if $(\exists q)\ q \rightarrow p \in B$ and $H \cup B \models q$.

Typically, there exist several hypotheses H explaining O. We call each of them a *candidate hypothesis*, and each literal in a hypothesis an *elemental hypothesis. Cost-based abduction* (CBA) is defined as abduction which identifies the minimum-cost explanation H^* among a set $\mathcal{H}$ of candidate explanations. Formally, we find $H^* = \arg\min_{H \in \mathcal{H}} cost(H)$, where *cost* is a function $\mathcal{H} \rightarrow \mathbb{R}$, which is called the *cost function.* Several kinds of cost functions have been proposed in prior work on cost-based abduction [1, 2, 19, 20, etc.]. For instance, Hobbs et al. [2] use a cost function that favors a fewest elemental hypotheses and a shorter proof path. The function is represented by the sum of costs of elemental hypotheses, where the cost of elemental hypothesis is in proportion to the distance from the observations on a proof graph.

2.2 ILP-based formulation of CBA

In this section, we briefly review Inoue and Inui [18]'s ILP formulation of cost-based abduction. The main idea is that explanation finding in CBA can be regarded as the weighted combinatorial optimization problem of literals. They thus formulate CBA as an ILP optimization problem, where the search space of CBA is represented as ILP variables and constraints, and the cost function is used as the ILP objective, in order to exploit the state-of-the-art combinatorial optimization technology in Operations Research.

Here we give an intuitive description of their approach, using the diagram illustrated in Figure 1. Given an abduction problem (i.e., background knowledge B and observations O), they first create set P of *potential elemental hypotheses*, a set of instantiated literals that are potentially included as constituents of explanations of O (i.e. Step 1 in Figure 1). This procedure is called the *search-space generation.* For enumerating potential elemental hypotheses, they apply *backward-chaining* with axioms in B, and instantiate the body of axioms. For instance, in Figure 1, we add two instantiated literals $s(y)$, $t(u)$ to P, which might be the explanations of $q(y) \in O$, performing backward-chaining on $q(y)$ with axiom $s(x) \wedge t(y) \rightarrow q(x)$. Using the set P, they represent the search space of explanations as an ILP optimization problem as follows.


Input:
B: $\{r(x) \rightarrow s(x), s(x) \wedge t(y) \rightarrow q(x)\}$
O: $\exists x, y\ q(y) \wedge r(A) \wedge s(x)$
Step 1. Generate P
Backward-chaining:
$s(y) \wedge t(u)$
$r(x)$
$y=x$
$A=x$
$\exists x, y\ q(y) \wedge r(A) \wedge s(x)$
Set of potential elemental hypotheses:
$P = \{s(y), t(u), r(x), q(y), r(A), s(x), x=y, A=x\}$
Step 2. Solve ILP optimization problem
Output:
H^*: $\exists x, y\ q(y) \wedge r(A) \wedge s(x) \wedge r(x) \wedge x=A$
ILP constraints:
C1: $h_{q(y)} = 1$
C2: $r_{s(x)} \leq h_{r(x)}$; $h_{s(y)}=h_{t(u)}$
C3: $2u_{r(x), r(A)} \leq h_{r(x)} + h_{r(A)}$
C4: $u_{r(x), r(A)} \leq s_{x,A}$
C5: $s_{x,A} + s_{x,B} \leq 1$
C6: $s_{y,A} - s_{x,A} - s_{x,y} \geq -1$

Candidate hypothesis:	ILP variables:										
	$h_{q(y)}$	$h_{r(A)}$	$h_{s(x)}$	$h_{s(y)}$	$h_{t(u)}$	$h_{r(x)}$	$s_{x,A}$	$s_{x,y}$	$s_{y,A}$	$u_{r(A),r(x)}$	$u_{s(x),s(y)}$
H_1: $q(y) \wedge r(A) \wedge s(x)$	1	1	1	0	0	0	0	0	0	0	0
H_2: $q(y) \wedge r(A) \wedge s(x) \wedge r(x)$	1	1	1	1	0	1	0	0	0	0	0
H_3: $q(y) \wedge r(A) \wedge s(x) \wedge r(x) \wedge x=A$	1	1	1	1	0	1	1	0	0	0	1
H_4: $q(y) \wedge r(A) \wedge s(x) \wedge r(x) \wedge s(y) \wedge t(u) \wedge x=A \wedge x=y$	1	1	1	1	1	1	1	1	1	1	1

Fig. 1. Summary of Inoue and Inui [18]'s ILP-based approach.

Hypothesis inclusion: For each $p \in P$, ILP variables $h_p \in \{0, 1\}$ are introduced to represent whether p is hypothesized ($h_p = 1$) or not ($h_p = 0$). For example, H_2 in Figure 1 holds $h_{r(x)} = 1$, where $r(x)$ is included in H_2.

Cost of hypothesis: To calculate $cost(H)$ of each candidate hypothesis H, they use the cost function defined in Hobbs et al. [2]'s weighted abduction. In weighted abduction, $cost(H)$ is represented by the sum of the costs for $p \in P$ such that p is hypothesized (i.e., $h_p = 1$) but not explained. They thus introduce another ILP variable $r \in \{0, 1\}$ for representing whether p is explained ($r_p = 1$) or not ($r_p = 0$). The final objective function of the ILP problem is given by: **min.** $cost(H) = \sum_{p \in \{p | p \in P, h_p = 1, r_p = 0\}} cost(p)$, where $cost(p)$ is the cost of a literal p. The cost of a literal is determined by the total unreliability of backward-inferences that are used for abducing the literal. This optimization amounts to Step 2 in Figure 1. They optimize this objective with the six types of ILP constraints as shown in Figure 1. In the rest of this section, we describe only two of them, which are necessary for the readers to follow the discussion below due to spatial limitations.

To handle first-order predicate logic, variable substitution must be taken into account to control the unification of elemental hypotheses. For representing the status of unification, they introduce another class of variables $u_{p,q} \in \{0, 1\}$ for each $p, q \in P$, which takes 1 if p is unified with q. Concerning variable substitution, another type of ILP variables s are introduced, where $s_{x,y} = 1$ if x is substituted for y, 0 otherwise. s is symmetric (i.e., $s_{x,y} = s_{y,x}$). In Figure 1, $u_{r(x),r(A)}$ and $s_{x,A}$ are introduced. In H_3, the variables $u_{r(x),r(A)}$, $s_{x,A}$ are set to 1 because $r(x)$ is unified with $r(A)$, and $x = A$ is assumed. Note that unification of $r(x)$ with $r(A)$ is allowed only if x are substituted with A. In addition, the substitution relation must be transitive (e.g. $y = A$ must hold if $x = y$ and $x = A$ hold). For keeping those consistency, they impose two ILP constraints:

Constraint 4[1]: Two literals $q(x_1, x_2, ..., x_n) \equiv q(\mathbf{x})$ and $q(y_1, y_2, ..., y_n) \equiv q(\mathbf{y})$ are allowed to be unified (i.e., $u_{q(\mathbf{x}),q(\mathbf{y})} = 1$) only if all variable substitutions x/y involved in the unification are activated (i.e., $s_{x_i,y_i} = 1$ for all $i \in \{1, 2, ..., n\}$). This can be expressed as:

$$n \cdot u_{q(\mathbf{x}),q(\mathbf{y})} \leq \sum_{i=1}^{n} s_{x_i,y_i} \tag{1}$$

In Figure 1, the constraint $u_{r(x),r(A)} \leq s_{x,A}$ is generated since x needs to be substituted for A when $r(x)$ and $r(A)$ are unified.

Constraint 6: s is transitive; namely $s_{x,z}$ must be 1 if $s_{x,y} = 1$ and $s_{y,z} = 1$. This can be expressed as the following constraints[2]:

$$s_{x,z} - s_{x,y} - s_{y,z} \geq -1 \tag{2}$$
$$-s_{x,z} + s_{x,y} - s_{y,z} \geq -1 \tag{3}$$
$$-s_{x,z} - s_{x,y} + s_{y,z} \geq -1 \tag{4}$$

They generate $O(n^3)$ transitivity constraints, where n is the number of logical terms. As the reader will see in Sec. 4, this makes inference intractable in large-scale inference. We propose how this drawback can be overcome by exploiting Cutting Plane Inference in Sec. 3.2.

3 Full-fledged first-order predicate logic abduction with cutting plane inference

In this section, we first present an extended formulation of Inoue and Inui [18] over full fledged first-order predicate logic. We then describe how to apply Cutting Plane Inference to best-explanation finding, for avoiding the intractability that arises from the extension and generation of transitivity constraints.

3.1 Handling negation for supporting full-fledged FOPL

The ILP formulation described in Sec. 2.2 does not provide full support for FOPL; it cannot represent negation. As a background knowledge, they accept only Horn clauses as axioms, and positive literals as observations. However, the capability of handling negations is crucial for a wide range of abductive reasoning. For example, in abduction-based natural language interpretation, one can imagine that it needs to handle negated expressions, such as *"I don't know."*

[1] The numbers of constraints correspond to the numbers presented in [18].

[2] Inoue and Inui [18] introduce the form of inequality $s_{x,y}+s_{y,z} \leq 2 \cdot s_{x,z}$ as transitivity constraints. However, this constraint does not appropriately represent transitivity; thus we replace them with inequalities (2)–(4).

In the rest of this section, we give two formulations for expressing negative literals (e.g. $\neg p$) and inequality of variables (e.g. $x \neq y$), for making Inoue and Inui [18]'s framework support full FOPL. By handling negative literals, we are able to assume the background knowledge to be a set of full-fledged first-order logical formulae represented in the clausal normal form, i.e. a set of implicitly skolemized disjunctive literals (e.g. $\{\neg p(x), q(x)\}$, $\{\neg p(x), q(f(x), x), r(f(x))\}$). To eliminate disjunctions from a clause, we convert each clause $\{L_1, \dots, L_n\}$ to a set of single literal-headed clauses of the form of $\neg L_1 \wedge \dots \wedge \neg L_{i-1} \wedge \neg L_{i+1} \wedge \dots \wedge \neg L_n \rightarrow L_i$ for each L_i. Henceforth, we call a clause of this form an *axiom*, the right hand side the *head*, and the left hand side the *body*.

First, consider the case where two literals $p(x)$ and $\neg p(y)$ are in set P of potential elemental hypotheses such that $p(x)$ and $p(y)$ are unifiable. We want to prohibit the two literals from being hypothesized simultaneously if x is substituted with y. One can imagine a simple inequality such as $h_{p(x)} + h_{\neg p(y)} \leq 1$, however, it is not enough because $p(x)$ and $p(y)$ can be both hypothesized (i.e. $h_{p(x)}$ and $h_{\neg p(y)}$ can be 1 simultaneously) if x is not substituted for y. This constraint can be correctly represented by incorporating the ILP variable $s_{x,y}$ which represents the variable substitution of x for y:

Proposed Constraint 1: Two literals $q(x_1, x_2, ..., x_n) \equiv q(\mathbf{x})$ and $\neg q(y_1, y_2, ..., y_n) \equiv \neg q(\mathbf{y})$ cannot be both hypothesized ($h_{q(\mathbf{x})} = 1$ and $h_{\neg q(\mathbf{y})} = 1$) if variable substitutions x_i/y_i are activated ($s_{x_i,y_i} = 1$) for all $i \in \{1, 2, ..., n\}$. This can be expressed as: $h_{q(\mathbf{x})} + h_{\neg q(\mathbf{y})} + \sum_{i=1}^{n} s_{x_i,y_i} \leq 1 + n$. Note that the case where $\mathbf{x} = \mathbf{y}$ reduces to: $h_{q(\mathbf{x})} + h_{\neg q(\mathbf{x})} \leq 1$. This type of constraint grows in $O(nm)$ for each predicate p, where n is the number of positive instantiation of p in P, and m is the number of negative instantiation of p in P.

The important question here is how to find the pair $q(\mathbf{x})$ and $\neg q(\mathbf{y})$. In order to find potential contradictions, one can perform forward reasoning. However, the problem is how to control the overall search process because chaining might be repeated infinitely. Terminating the search at a certain depth could miss potential contradictions. Given $q(\mathbf{x})$ and $r(\mathbf{x})$ as potential elemental hypotheses, for example, we may fail to find that $\neg q(\mathbf{x})$ could be derived from $r(\mathbf{x})$ in several forward-chaining steps. This is a long-standing problem in logic-based reasoning. One can address this problem by adopting some heuristics such as A* search.

We now describe how the inequality of variables, where two variables are prohibited to unify due to some constraints, can be formulated. This kind of constraint is also important for abductive inference. For example, imagine natural language interpretation systems. Given the sentence "*A girl sent a present to another girl*", it is desirable to express that the two girls must not be identical. Such a constraint can be expressed straightforwardly in the ILP formulation:

Proposed Constraint 2: For each pair of (existentially quantified) variables x and y in set P of potential elemental hypotheses that must not be identical (i.e. $x \neq y$), introduce the following equality:

$$s_{x,y} = 0. \tag{5}$$

In our experiments, we use the six types of constraints described in Sec. 2.2 and two constraints newly introduced above.

Algorithm 1 CPI4CBA(Background Knowledge **B**, Observation **O**)

```
1: $(\Psi, I) \leftarrow$ createBaseILP$(B, O)$
2: repeat
3:   $S \leftarrow$ solveILP$(\Psi, I)$; $V \leftarrow \{\}$
4:   for $(x, y) \in$ unifiedTerms$(S)$ do
5:     for $z \in$ termsUnifiableWith$(x) \cup$ termsUnifiableWith$(y)$ do
6:       if $(s_{x,z} = 0$ and $s_{y,z} = 1)$ or $(s_{x,z} = 1$ or $s_{y,z} = 0)$ then
7:         $V \leftarrow V \cup \{-s_{x,y} - s_{x,z} + s_{y,z} \geq -1, -s_{x,y} + s_{x,z} - s_{y,z} \geq -1\}$
8:       end if
9:     end for
10:  end for
11:  $I \leftarrow I \cup V$
12: until $V \neq \phi$
```

3.2 Cutting plane inference for CBA

The major drawback of the ILP formulation is that it needs to generate $O(n^3)$ transitivity constraints, where n is the number of logical terms, because we perform inference over FOPL-based representation. That makes inference intractable (see Sec. 4 for empirical evidence) because it generates an ILP optimization problem that has quite a large number of constraints. Moreover, handling negation quadratically increases Proposed Constraint 1.

How do we overcome this drawback? The idea is that "all the transitivity constraints may not be violated all at once; so we gradually optimize and add transitivity constraints *if violated* in an iterative manner." More formally, we propose to apply *Cutting Plane Inference (CPI)* to the CBA problems. CPI is an exact inference optimization technique that is originally developed for solving large linear programming (LP) problems in Operations Research [21]. CPI has been successfully applied to a wide range of constrained optimization problems where constraints are very large [22–25, etc.], from probabilistic deductive inference problems [23] to machine learning problems [24]. To the best of our knowledge, however, our work is the first successful work to apply CPI to abductive reasoning tasks. In principle, CPI solves optimization problem in an iterative manner as follows: it solves an optimization problem without constraints, and then adds violated constraints to the optimization problem. When the iteration terminates, it guarantees solutions to be optimal. The proposed algorithm, called *CPI4CBA*, is also an exact inference framework.

How do we apply the technique of CPI to cost-based abduction problems? Intuitively, we iterate the following two steps: (i) solving an abduction problem without enforcing transitivity on logical atomic terms, and (ii) generating transitivity constraints dynamically when transitiveness of unification is violated (e.g. $x = y \wedge y = z \wedge z \neq x$). The iteration terminates if there is no violated unification transitivity. The pseudo-code is given in Algorithm 1. In line 1, we first create an ILP optimization problem described in Sec. 2.2 and Sec. 3.1 but without transitivity constraints (i.e. Constraint 6), where Ψ denotes a set of ILP variables, and I denotes a set of ILP constraints. In line 2–12, we repeat: checking consistency of unification transitiveness, adding constraints for violated transitiveness, and re-optimizing. In line 3, we find the solution S for the current ILP optimization problem. Then, for each pair (x, y) of logical atomic terms unified in the solution S (line 4), find the logical term z which is unifiable with x or y (line 5). If the transitive relation x, y with respect to z is violated (i.e. $s_{x,z} = 0 \wedge s_{y,z} = 1$ or $s_{x,z} = 1 \wedge s_{y,z} = 0$), then we generate constraints for preventing this violation, and keep it in set V of constraints (line 6–8). Finally, we again perform an ILP optimization with newly generated constraints (line 11 and 3). The iteration ends when there is no violated transitiveness (line 12).

The key advantages of CPI4CBA is that it can reduce the time of search-space generation, and it is also expected to reduce the time of ILP optimization. CPI4CBA does not generate all the transitivity constraints before optimization, which saves the time for search-space generation. In addition, optimization problems that we solve would become smaller than the original problem in most cases, because not all the transitivity constraints may not be necessary to be considered. In the worst case, we need to solve the optimization problem that is same as the original one; but in most cases we found out that we do not need to. We will show its empirical evidence through large-scale evaluation in Sec. 4.

4 Runtime Evaluation

How much does CPI improve the *runtime* of ILP-based reasoner? Does CPI scale to larger real-life problems? To answer these questions, we evaluated the CPI4CBA algorithm in two settings: (i) **STORY**, the task of plan recognition, and (ii) **RTE**, the popular, knowledge-intensive, real-life natural language processing task of *Recognizing Textual Entailment* (RTE). While most of the existing abductive reasoning systems are evaluated on rather small, and/or artificial datasets [26–28, etc.], our evaluation takes a real-life, much larger datasets (see Sec. 4.1). In our experiments, we compare our system with: (i) Inoue and Inui's formulation [18], and (ii) the systems [26, 28, 29] based on Markov Logic Networks (MLNs) [30]. For our experiments, we have used a 12-Core Opteron 6174 (2.2GHz) 128 GB RAM machine. We used Gurobi Optimizer[3], which is an efficient ILP solver. It is commercial but an academic license is freely available.

4.1 Settings

STORY: For this setting, we have used Ng and Mooney [31]'s story understanding dataset, which is widely used for evaluation of abductive plan recognition systems [26–28]. In this task, we need to abductively infer the top-level plans of characters from actions which are represented by the logical forms (e.g. $\mathit{getting\_off}(\mathit{Getoff16}) \wedge \mathit{agent\_get\_off}(\mathit{Getoff16}, \mathit{Fred16}) \wedge \mathit{name}(\mathit{Fred16}, \mathit{Fred})$). The dataset consists of 50 plan recognition problems and 107 background Horn

[3] http://www.gurobi.com/

clauses (e.g. $\mathit{go\_step}(r, g) \wedge \mathit{going}(g) \rightarrow \mathit{robbing}(r)$). The dataset contains on average 12.6 literals in observed logical forms. To make the predicates representing top-level plans (e.g. shopping, robbing) disjoint, we generated 73 disjointness axiom by using the formulation[4] described in Sec. 3.1. Note that in Inoue and Inui [18]'s evaluation, disjointness constraints are not used. Regarding a cost function, we followed Hobbs et al. [2]'s weighted abduction theory. For each axiom, we have set the sum of the axiom weights equal to 1.2 (e.g. $\mathit{inst\_shopping}(s)^{0.6} \wedge \mathit{store}(t, s)^{0.6} \rightarrow \mathit{shopping\_place}(t)$).

RTE: For observations (input), we employed the second challenge of RTE dataset[5]. In the task of RTE, we need to correctly determine whether one text (called *text*, or T) entails another (called *hypothesis*, or H) or not. The dataset consists of development set and test set, each of which includes 800 natural language text-hypothesis pairs. We have used all of the 800 texts from test set. We have converted texts into logical forms presented in [32] using the Boxer semantic parser [33]. The number of literals in observations is 29.6 literals on average. For background knowledge, we have extracted 289,655 axioms[6] from WordNet 3.0 [5], and 7,558 axioms from FrameNet 1.5 [6] following [10]. In principle, the WordNet knowledge base contains several kinds of lexical relations between words, such as IS-A, ontological relations (e.g. $\mathit{dog}(x) \rightarrow \mathit{animal}(x)$). FrameNet knowledge bases contain lexeme-to-frame mappings, frame-frame relations, etc. For example, the mapping from surface realization "give to" to a frame "Giving" is given by: $\mathit{Giving}(e_1, x_1, x_2, x_3)^{1.3} \wedge \mathit{donor}(e_1, x_1)^{0.1} \wedge \mathit{recipient}(e_1, x_2)^{0.2} \wedge \mathit{theme}(e_1, x_3)^{0.1} \rightarrow \mathit{give}(e_1, x_1, x_3) \wedge \mathit{to}(e_2, e_1, x_2)$. We again followed Hobbs et al. [2]'s weighted abduction theory for calculating the cost of hypothesis. We calculated the costs by following Ovchinnikova et al. [10] in this setting.

4.2 Results and discussion

The reasoner was given a 2-minute time limit for each inference step (i.e. search-space generation and ILP optimization). In Table 1, we show the results of each setting for two inference method in Table 1: (i) *IAICBA*: the inference method without CPI (i.e. Inoue and Inui [18]'s formulation with proposed constraints 1 and 2), and (ii) *CPI4CBA*: inference method with CPI (i.e. our proposal). In order to investigate the relation between the size of search space and the runtime, we show the results for each depth, which we used for limiting the length of backward-chaining. In the "Generation" column, we show the runtime that is taken for search-space generation in seconds averaged over all problems whose search-space generation is finished within 2 minutes. In the parenthesis, we show the percentage of those problems. In the column "ILP inf", we show the runtime of ILP optimization averaged on only problems such that both

[4] For example, we generate $h_{robbing(x)} + h_{shopping(y)} + s_{x,y} \leq 2$.

[5] http://pascallin.ecs.soton.ac.uk/Challenges/RTE2/

[6] Extracted relations are: word-to-synset mapping, hypernym-hyponym, cause-effect, entailment, derivational, instance-of relations.

Setting	Method	Depth	Generation [sec.]	ILP inf [sec.]	# of ILP cnstr
STORY	IAICBA	1	0.02 (100.0 %)	0.60 (100.0 %)	3,708
		2	0.12 (100.0 %)	5.34 (100.0 %)	23,543
		3	0.33 (100.0 %)	8.11 (100.0 %)	50,667
		∞	0.35 (100.0 %)	9.00 (100.0 %)	61,122
	CPI4CBA	1	0.01 (100.0 %)	0.34 (100.0 %)	784 ($\mathit{\Delta}$ 451)
		2	0.07 (100.0 %)	4.15 (100.0 %)	7,393 ($\mathit{\Delta}$ 922)
		3	0.16 (100.0 %)	3.36 (100.0 %)	16,959 ($\mathit{\Delta}$ 495)
		∞	**0.22 (100.0 %)**	**5.95 (100.0 %)**	24,759 ($\mathit{\Delta}$ 522)
RTE	IAICBA	1	0.01 (100.0 %)	0.25 (99.7 %)	1,104
		2	0.08 (100.0 %)	2.15 (98.1 %)	5,185
		3	0.56 (99.9 %)	5.66 (93.0 %)	16,992
		∞	4.78 (90.7 %)	15.40 (60.7 %)	36,773
	CPI4CBA	1	0.01 (100.0 %)	0.05 (100.0 %)	269 ($\mathit{\Delta}$ 62)
		2	0.04 (100.0 %)	0.35 (99.6 %)	1,228 ($\mathit{\Delta}$ 151)
		3	0.09 (100.0 %)	1.66 (99.0 %)	2,705 ($\mathit{\Delta}$ 216)
		∞	**0.84 (98.4 %)**	**11.73 (76.9 %)**	10,060 ($\mathit{\Delta}$ 137)

Table 1. The results of averaged inference time in **STORY** and **RTE**.

search-space generation and ILP optimization are finished within 2 minutes, as well as the percentage of those problems (e.g. 80 % means "for 80 % of all the problems, search-space generation was finished within 2 minutes, and so was ILP inference."). In the "# of ILP cnstr" column, we show the averaged number of generated ILP constraints. Concerning CPI4CBA, the number denotes the total number of constraints considered in the end, including the constraints added by CPI. The number marked by $\mathit{\Delta}$ indicates the number of constraints that are added during CPI (i.e. how many times line 7 in Algorithm 1 executed).

Overall, the runtimes in both search-space generation and ILP inference are dramatically improved from IAICBA to CPI4CBA in both settings, as shown in Table 1. In addition, CPI4CBA can find optimal solutions in ILP inference for more than 76 % of the problems, even for depth ∞. This indicates that CPI4CBA scales to larger problems. From the results of IAICBA in **RTE** settings, we can see the significant bottleneck of Inoue and Inui [18]'s formulation in large-scale reasoning: the time of search-space generation. The search-space generation could be done within 2 minutes for only 90.7 % of the problems. CPI4CBA successfully overcomes this bottleneck. CPI4CBA is clearly advantageous in the search-space generation because it is not necessary to generate transitivity constraints, an operation that grows cubically before optimization.

In addition, CPI4CBA reduces the time of ILP inference significantly. In ILP inference, CPI did not guarantee the reduction of inference time in theory; *however*, as shown in Table 1, we found that the number of ILP constraints actually used is much less than the original problem. CPI4CBA successfully reduces the complexity of the ILP optimization problems in practice. This is also supported by the fact that CPI4CBA keeps 76.9% in "ILP inf" for Depth

$= \infty$ because it solves very large ILP optimization problems that fail to be generated in IAICBA.

Finally, we compare our results with other existing systems. Overall, the presented reasoner is dramatically faster than the other systems. First, we immediately see that the proposed method is more efficient than Inoue and Inui [18]'s formulation (i.e. IAICBA). Regarding the MLN-based systems [26, 28, 29], our results are comparable, or more efficient than the existing systems. For the **STORY** setting, Singla and Mooney [28] report the results of two systems with an exact inference technique using CPI for MLNs [23]: (i) Kate and Mooney [26]'s approach: 2.93 seconds, and (ii) Singla and Mooney [28]'s approach: 0.93 seconds[7]. MLN-based approaches seem to be reasonably efficient for small datasets. However, it does not scale to larger problems; for the **RTE** setting, Blythe et al. [29] report that only 28 from 100 selected RTE-2 problems could be run to completion. The processing time was 7.5 minutes on average [personal communication]. On the other hand, our method solves 76.9% of all the problems, where suboptimal solutions are still available for the rest of 21.5%, and it takes only 0.84 seconds for search-space generation, and 11.73 seconds for ILP inference.

5 Related work

A number of methods attempting to efficiently find the best explanation have been proposed [17, 15, 34, 35, 16, 18, etc.]; however, most of them focus on improving the inefficiency of propositional logic-based abduction. Although propositionalization techniques are available for applying these methods to FOPL abduction, it will lead to an exponential growth of ground instances. Hence they would not scale to larger problems for FOPL abduction with large KBs.

Recently, Markov Logic Networks (MLNs) [30] are used for emulating abduction [26, 29, 28, etc.]. They provide full support of first-order predicate logic; however, MLN-based approaches require special procedures to convert abduction problems into deduction problems because of the deductive nature of MLNs. The pioneering work of MLN-based abduction [26] converts background axioms 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: $q \rightarrow p_1 \vee p_2 \vee p_3$, $q \rightarrow \neg p_1 \vee \neg p_2$, $q \rightarrow \neg p_1 \vee \neg p_3$ etc.). MLN-based approaches suffer from the inefficiency of inference due to the increase of converted axioms.

One important work for FOPL abduction is Inoue and Inui [18]'s approach formulating first-order predicate logic abduction as an ILP optimization problem. However, as mentioned in Sec. 1 and Sec. 2.2, this formulation has two significant drawbacks for large-scale reasoning on real-life problems: (i) the combinatorial growth of transitivity constraints which arises from support for FOPL (see Sec. 3.2), (ii) negation is not supported.

[7] This is the result of MLN-HC in [28]. MLN-HCAM cannot be directly compared with our results, since the search space is different from our experiments because they unify some assumptions in advance to reduce the search space.

6 Conclusion

We have proposed an ILP-based formulation for cost-based abduction in full-fledged first-order predicate logic, extending Inoue and Inui [18]'s formulation. Compared to prior work, our method is more expressive and efficient. Although full-fledged FOPL reasoning is computationally expensive, our proposed optimization strategy CPI brings us to a significant boosting of the efficiency of the reasoner. We have evaluated our method on two datasets, including real-life problems (i.e. RTE dataset with axioms generated from WordNet and FrameNet). Our evaluation revealed that our inference method CPI4CBA was highly efficient than other existing systems. In future work, we will develop methods for automatic tuning of costs of elemental hypotheses. Specifically, we plan to represent the cost function as a weighted linear feature function, and then apply a standard linear training algorithm such as perceptrons. Also, we will evaluate the abduction-based framework in terms of the prediction accuracy on real-life tasks. We intend to apply abduction to co-reference resolution, the task of identifying referential relations in natural language texts, as a first step. The abductive inference engine presented in this paper is made publicly available.

Acknowledgments

This work was partially supported by Grant-in-Aid for JSPS Fellows (22-9719) and Grant-in-Aid for Scientific Research (23240018). The authors would like to thank the reviewers for their insightful comments.

References

1. Charniak, E., Goldman, R.P.: A Probabilistic Model of Plan Recognition. In: AAAI. (1991) 160–165
2. Hobbs, J.R., Stickel, M., Martin, P., Edwards, D.: Interpretation as abduction. Artificial Intelligence **63** (1993) 69–142
3. Shanahan, M.: Perception as Abduction: Turning Sensor Data Into Meaningful Representation. Cognitive Science **29**(1) (2005) 103–134
4. Peraldi, S.E., Kaya, A., Melzer, S., Möller, R., Wessel, M.: Multimedia Interpretation as Abduction. In: Int'l Workshop on Description Logics. (2007)
5. Fellbaum, C., ed.: WordNet: an electronic lexical database. MIT Press (1998)
6. Ruppenhofer, J., Ellsworth, M., Petruck, M., Johnson, C., Scheffczyk, J.: FrameNet II: Extended Theory and Practice. Technical report, Berkeley, USA (2010)
7. Chambers, N., Jurafsky, D.: Unsupervised Learning of Narrative Schemas and their Participants. In: ACL. (2009) 602–610
8. Schoenmackers, S., Davis, J., Etzioni, O., Weld, D.: Learning First-order Horn Clauses from Web Text. In: EMNLP. (2010) 1088–1098
9. Hovy, D., Zhang, C., Hovy, E., Penas, A.: Unsupervised discovery of domain-specific knowledge from text. In: ACL. (2011) 1466–1475
10. Ovchinnikova, E., Montazeri, N., Alexandrov, T., Hobbs, J.R., McCord, M., Mulkar-Mehta, R.: Abductive Reasoning with a Large Knowledge Base for Discourse Processing. In: IWCS, Oxford, UK (2011) 225–234

11. Dagan, I., Dolan, B., Magnini, B., Roth, D.: Recognizing textual entailment: Rational, evaluation and approaches - Erratum. NLE **16**(1) (2010) 105
12. Bylander, T., Allemang, D., Tanner, M.C., Josephson, J.R.: The computational complexity of abduction. Artificial Intelligence **49**(1-3) (may 1991) 25–60
13. Mulkar, R., Hobbs, J., Hovy, E.: Learning from Reading Syntactically Complex Biology Texts. In: Commonsense, Palo Alto (2007)
14. Poole, D.: Logic Programming, Abduction and Probability: a top-down anytime algorithm for estimating prior and posterior probabilities. New Generation Computing **11(3-4)** (1993) 377–400
15. Ishizuka, M., Matsuo, Y.: SL Method for Computing a Near-optimal Solution using Linear and Non-linear Programming in Cost-based Hypothetical Reasoning. In: PRCAI. (1998) 611–625
16. Chivers, S.T., Tagliarini, G.A., Abdelbar, A.M.: An Evolutionary Optimization Approach to Cost-Based Abduction, with Comparison to PSO. In: IJCNN. (2007) 2926–2930
17. Santos, E.: A linear constraint satisfaction approach to cost-based abduction. Artificial Intelligence **65 (1)** (1994) 1–27
18. Inoue, N., Inui, K.: ILP-Based Reasoning for Weighted Abduction. In: AAAI Workshop on Plan, Activity and Intent Recognition. (2011)
19. Raina, R., Ng, A.Y., Manning, C.D.: Robust textual inference via learning and abductive reasoning. In: AAAI. (2005)
20. Stern, A., Dagan, I.: A confidence model for syntactically-motivated entailment proofs. In: ACL. (2011) 455–462
21. Dantzig, G.B., Fulkerson, R., Johnson, S.M.: Solution of a large-scale traveling salesman problem. Operations Research **2**(4) (1954) 393–410
22. Riedel, S., Clarke, J.: Incremental Integer Linear Programming for Non-projective Dependency Parsing. In: EMNLP. (2006) 129–137
23. Riedel, S.: Improving the Accuracy and Efficiency of MAP Inference for Markov Logic. In: UAI. (2008) 468–475
24. T. Joachims, T. Finley, C.J.Y.: Cutting-plane training of structural svms. In: Machine Learning. (2009) 27–59
25. J. Berant, T.A., Goldberger, J.: Global Learning of Typed Entailment Rules. In: ACL. (2008) 610–619
26. Kate, R.J., Mooney, R.J.: Probabilistic Abduction using Markov Logic Networks. In: PAIRS. (2009)
27. Raghavan, S., Mooney, R.J.: Bayesian Abductive Logic Programs. In: STARAI. (2010) 82–87
28. Singla, P., Domingos, P.: Abductive Markov Logic for Plan Recognition. In: AAAI. (2011) 1069–1075
29. Blythe, J., Hobbs, J.R., Domingos, P., Kate, R.J., Mooney, R.J.: Implementing Weighted Abduction in Markov Logic. In: IWCS, Oxford, UK (2011) 55–64
30. Richardson, M., Domingos, P.: Markov logic networks. ML (2006) 107–136
31. Ng, H.T., Mooney, R.J.: Abductive Plan Recognition and Diagnosis: A Comprehensive Empirical Evaluation. In: KR. (1992) 499–508
32. Hobbs, J.R.: Ontological promiscuity. In: ACL, Chicago, Illinois (1985) 61–69
33. Bos, J.: Wide-Coverage Semantic Analysis with Boxer. In: STEP. (2008) 277–286
34. Prendinger, H., Ishizuka, M.: First-Order Diagnosis by Propositional Reasoning: A Representation-Based Approach. In: DX. (1999) 220–225
35. Abdelbar, A.M., Hefny, M.: An efficient lp-based admissible heuristic for cost-based abduction. JETAI **17**(3) (2005) 297–303