Extending ILP-based Abductive Reasoning with Cutting Plane Inference

井之上 直也，乾 健太郎

東北大学

アブダクション (Abduction)

- 観察に対する最良の説明 (仮説) を求める推論

観察 (Observation):

John got a gun ***John went to a store***

背景知識 (Background Knowledge):

go-hunt → get-gun
go-shopping → go-to-store
rob → get-gun & go-to-store

説明 (Explanation):

John will be robbing?
John goes hunting and shopping?

アブダクション (Abduction)

- 観察に対する最良の説明 (仮説) を求める推論

観察 (Observation):

$$\textit{get-gun}(\textit{John}) \;\wedge\; \textit{go-to-store}(\textit{John})$$

背景知識 (Background Knowledge):

$$(\forall x)\ \textit{go-hunt}(x) \rightarrow \textit{get-gun}(x)$$
$$(\forall x)\ \textit{go-shopping}(x) \rightarrow \textit{go-to-store}(x)$$
$$(\forall x)\ \textit{rob}(x) \rightarrow \textit{get-gun} \wedge \textit{go-to-store}(x)$$

説明 (Explanation):

$$\textit{robbing}(\textit{John})$$
$$\textit{hunting}(\textit{John}) \wedge \textit{shopping}(\textit{John})$$

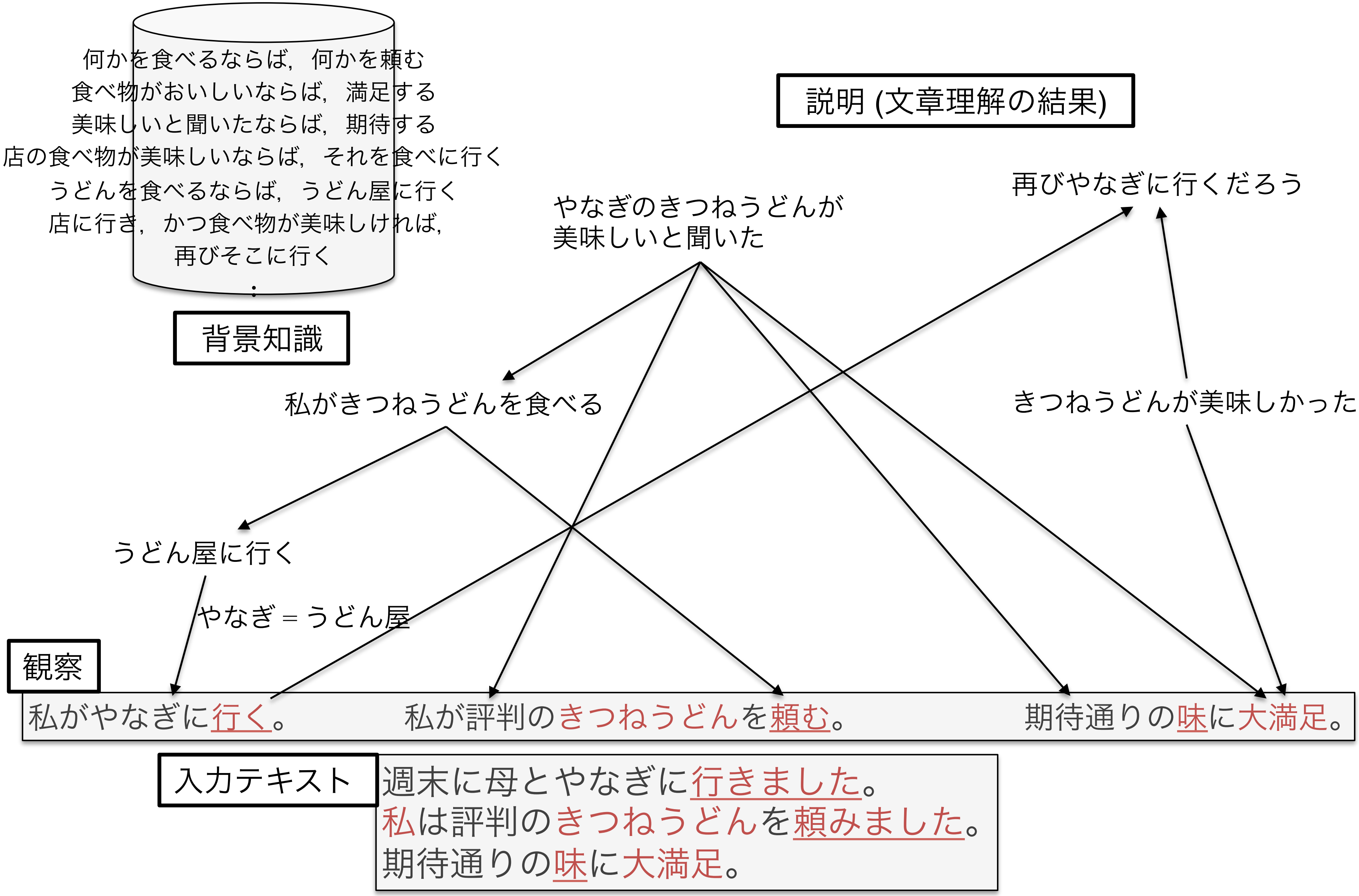
アブダクションによる談話解析

- ゴール: 談話理解の一般的な枠組みを作る
  - 談話解析の各コンポーネント (照応，談話構造の理解など) を統合し，結合推論できる枠組みを作りたい
  - 談話に潜在する情報 (意図など) を顕在化したい
  - 世界知識を談話理解に有効利用したい
- Interpretation as Abduction (IA) にもとづく談話理解モデルの構築に取り組んでいる

"Interpreting sentences is to find the lowest-cost explanation to the sentence."

--- Hobbs+ [93]

談話の最良の説明を求める = 談話を理解する



C. 効率的な説明選択の実現

(組み合わせ最適化問題であるため)
→ 😐 整数線形計画法による定式化 [Inoue & Inui 11]
→ 本研究はこの枠組みを拡張する

めの課題

評価関数の学習
伝播による学習
2], 若手 (発表3.)

$$\underset{E \in \mathcal{E}_O}{\arg\min}\ cost(E; \theta)$$

$\mathcal{E}_O$: 説明候補の集合
$cost$: 説明評価関数

A. 多様な説明候補を生成するための背景知識ベースの構築
→ ☺ 大規模知識獲得技術の発展

- ✓ イントロダクション
- □ 整数線形計画問題 (ILP) による
アブダクションの定式化
- □ Cutting Plane Inference
による高速化
- □ 評価実験

ILP によるアブダクションの定式化

アブダクション:
観測 O を最も良く説明する仮説 H^* を,
背景知識 B から求める推論

1. 仮説候補生成: 最良仮説の候補 H の集合 $\mathcal{H}$ を生成する
   - 観測 O を含意せよ ($B \cup H \models O$)
   - 矛盾するな ($B \cup H \not\models \perp$)
2. 仮説選択: 最良の仮説 H^* を選択する
   - コストが最小となる仮説を選べ
     ($H^* = \arg\min_{H \in \mathcal{H}} cost(H)$)

ILP によるアブダクションの定式化

→ **0-1 ILP 変数への値割り当ての組み合わせ**

ン:
る仮説 H^* を,
背景知識 ら求める推論

1. 仮説候補生成: 最良仮説の候補 H 生成する

→ **ILP 変数間の制約**

   – 観測 O を含意せよ ($B \cup H \models O$)
   – 矛盾するな ($B \cup H \not\models \perp$)

2. 仮説選択: 最良の仮説 H^* を選択する
   – コストが最小となる仮説を選べ

   ($H^* = \arg\min_{H \in \mathcal{H}} cost(H)$)

→ **ILP 目的関数**

☞ **ILP 変数**
ILP 目的関数
ILP 制約

入力

背景知識 B:

sick(x) → resign(x, y)
hate(x, y) → resign(x, y)
old(x) → resign(x, y)

sick(x) → go(x, y) ∧ hospital(y)
hospital(AbcHospital).
boring(y) → hate(x, y)

観測 O:

resign(Steve, Microsoft) ∧ ∃m go(m, AbcHospital)

ILP による定式化

潜在仮説集合 P

resign(Steve, Microsoft) go(m, AbcHospital)
sick(Steve) hate(Steve, Microsoft) old(Steve) sick(m) hospital(AbcHospital)
m=Steve boring(Microsoft) hospital(AbcHospital)

$cost(H) =$

出力

最良の説明 H^* ($B \cup H^* \models O$):

入力

☞ **ILP 変数**
ILP 目的関数
ILP 制約

背景知識 B:

sick(x) → resign(x, y)
hate(x, y) → resign(x, y)
old(x) → resign(x, y)

sick(x) → go(x, y) ∧ hospital(y)
hospital(AbcHospital).
boring(y) → hate(x, y)

観測 O:

resign(Steve, Microsoft) ∧ ∃m go(m, AbcHospital)

ILP による定式化

$h_{\text{resign(Steve, Microsoft)}} = 1$, $h_{\text{go(Steve, AbcHospital)}} = 1$

$h_{\text{sick(Steve)}} = 1$, $h_{\text{hate(Steve, Microsoft)}} = 0$, $h_{\text{old(Steve)}} = 0$, $h_{\text{sick(m)}} = 1$, $h_{\text{hospital(AbcHospital)}} = 0$

$s_{\text{m,Steve}} = 1$, $h_{\text{boring(Microsoft)}} = 0$, $h_{\text{hospital(AbcHospital)}} = 0$

$$cost(H) =$$

出力

最良の説明 H^* ($B \cup H^* \models O$):

sick(Steve) ∧ sick(m) ∧ Steve=m ∧ resign(Steve, Microsoft) ∧ go(Steve, AbcHospital)

入力

ILP 変数
☞ **ILP 目的関数**
ILP 制約

背景知識 B:

sick(x) → resign(x, y)
hate(x, y) → resign(x, y)
old(x) → resign(x, y)

sick(x) → go(x, y) ∧ hospital(y)
hospital(AbcHospital).
boring(y) → hate(x, y)

観測 O:

resign

ILP によ

説明の評価関数
新しい仮説を推論 (h=1): コスト増
他の仮説により説明 or 単一化 (r=1):コスト減

resign(S
go(Steve, AbcHospital)

$h_{sick(Steve)} = 1$, $h_{hate(Steve, Microsoft)}$ … $(Steve) = 0$, $h_{sick(m)} = 0$, $h_{hospital(AbcHospital)} = 0$

$s_{m,Steve} = 0$, $h_{boring(Microsoft)} = 0$, $h_{hospital(AbcHospital)} = 0$

$$cost(H) = \sum_{p \in P} h_p\, cost(p) - r_p\, reward(p)$$

出力

最良の説明 H^* $(B \cup H^* \models O)$:

sick(Steve) ∧ resign(Steve, Microsoft) ∧ go(Steve, AbcHospital)

入力

ILP 変
ILP 目
☞ **ILP 制**

背景知識 B:

sick(x) → resign(x, y)
hate(x, y) → resign(x, y)
sick(x) → go(x, y) ∧ hospital(y)
hospital(AbcHospital).
boring(y) → hate(x, y)

2. 仮説間の含意関係
e.g., $h_{\text{boring(Mi)}} \leq h_{\text{hate(St, Mi)}}$

1. 論理変数の等価関係は推移律を満たす
e.g., $s_{m,St} = 1 \land s_{m,u} = 1 \Rightarrow s_{St,u} = 1$

ft) ∧ ∃m go(m, AbcHospital)

3. 報酬を受け取る条件
e.g., $r_{\text{hate(St, Mi)}} \leq h_{\text{boring(Mi)}}$

$h_{\text{sick(Steve)}} = 1$, $h_{\text{hate(Steve, Microsoft)}} = 0$, $h_{\text{old(Steve)}} = 0$, $h_{\text{hospital(AbcHospital)}} = 0$
$s_{m,\text{Steve}} = 0$, $h_{\text{boring(Microsoft)}} = 0$, $h_{\text{hospital(AbcHospital)}} = 0$

$$cost(H) = \sum_{p \in P} h_p\, cost(p) - r_p\, reward(p)$$

出力

最良の説明 H^* ($B \cup H^* \models O$):
sick(Steve) ∧ resign(Steve, Microsoft) ∧ go(Steve, AbcHospital)

ILP による定式化の問題点

- 論理変数の等式間の推移律制約が多項式オーダで増加
  - 論理変数の組み合わせ (x, y, z) について
    - $x{=}y \wedge y{=}z \Rightarrow x{=}z$
    - $y{=}z \wedge x{=}z \Rightarrow x{=}y$
    - $x{=}z \wedge x{=}y \Rightarrow y{=}z$
- 長い文章を入力したときに大きな問題となる
  - 論理変数の数 = 文章における mention の数
- ILP 最適化問題を解く段階になかなか到達できない
  → ILP の準最適解も得られない

Cutting Plane Inference (CPI) の適用

- ボトルネック解消のアイデア
  - 逐次最適化法: Cutting Plane Inference の応用
  - 大規模な制約付き最適化問題を解く一テクニック:

    制約なしの状態で最適化
    →満たされぬ制約を追加→最適化
    →満たされぬ制約を追加→最適化
    →... を繰り返しながら最適化する手法

  - 推移律制約に対して Cutting Plane Inference を適用

Cutting Plane Inference (CPI) の適用

- 適用例
  - 1回目: 最良の説明: $x=y$, $y=z$, $x \neq z$
    - $x=y \wedge y=z \Rightarrow x=z$ を制約として追加
  - 2回目: 最良の説明: $x=y$, $y \neq z$, $x=z$
    - $x=z \wedge x=y \Rightarrow y=z$を制約として追加
  - 3回目: 最良の説明: $x=y$, $y=z$, $x=z$
    - すべての制約を満たしているので最適化終了
    - 最適化に必要な制約は 2/3 で済んだ!

推論時間の評価実験

- CPI はアブダクションをどれほど高速化できるだろうか?
- 実験データセット: *Recognizing Textual Entailment*
  - RTE2 開発セットを論理式に変換
  - 入力: 平均30リテラル x 800問
  - 知識:
    - 289,655 個の WordNet axioms (e.g. synset9(x) => synset10(x))
    - 7,558 個の FrameNet axioms
      (e.g. GIVING(e1, x, y) => GETTING(e2, y, z))
- 実行環境
  - ILP ソルバー: Gurobi optimizer 5.0
  - semantic parser: Boxer [Bos 08]

結果 (1): 全体の推論時間の変化

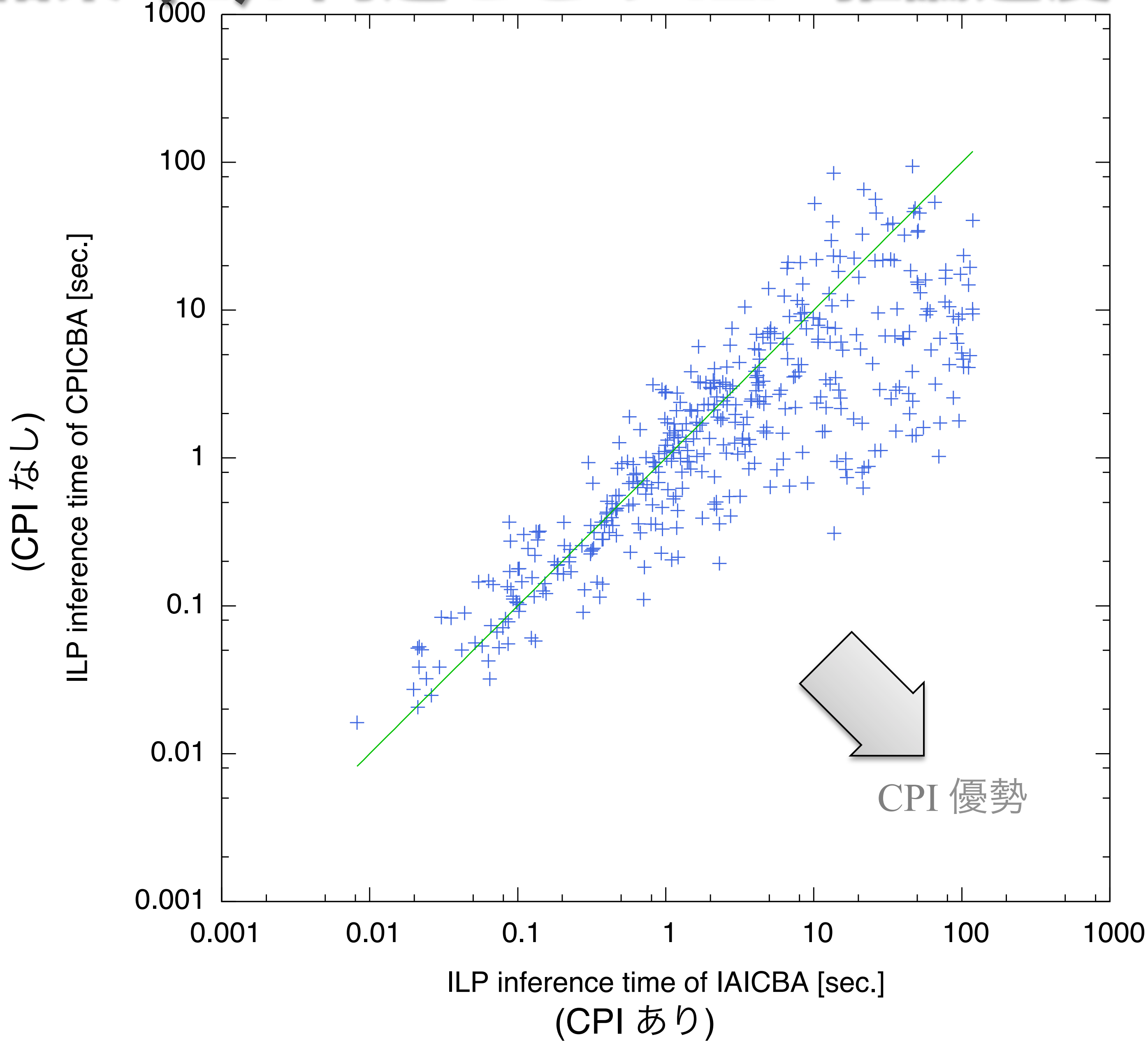
Method	Depth	Generation [sec.] (timeout = 120)
CPI なし	1	0.01 (100.0 %)
	2	0.08 (100.0 %)
	3	0.56 (99.9 %)
	∞	4.78 (90.7 %)
CPI あり	1	0.01 (100.0 %)
	2	0.04 (100.0 %)
	3	0.09 (100.0 %)
	∞	**0.84 (98.4 %)**

ILP 最適化問題の生成
時間を大幅に短縮できた

実際に考慮された制約数は大幅に
少なかった

- ILP 推論の時間も短縮
- 最適解を得られた問題数も増えた

結果 (2): 問題ごとの ILP 推論速度の変化



CPI その後

- コスト関数の学習が実用的な規模で実現可能に!
- CPI ベースの推論エンジンを用いた談話解析の結果を，共参照解析の評価セットで評価してみた!
  - データセット: CoNLL Shared Task 2011，303 documents
  - 1 document は，RTE のそれよりかなり長い
  - 知識ベース: WordNet, FrameNet, Narrative Schemas [Chambers & Jurafsky 07, 08, 09]
  - コスト関数のパラメタはデータから自動学習

System	BLANC-R	BLANC-P	BLANC-F
Abduction	59.9	60.9	60.3
Standford CoreNLP	**63.5**	**76.2**	**66.7**

  - 改良を重ね，徐々に state-of-the-art に近づいている
- ツールはこちら (旧バージョン，CPI 実装版は近日公開)
  - http://code.google.com/p/henry-tacitus/

まとめと今後の課題

- アブダクションによる談話解析の実現を目指して，推論効率の問題に取り組んだ
- 推論効率のボトルネックは，論理変数間の推移律制約が多項式オーダで増えることであった
- 推移律制約を逐次的に追加して最適化することにより，スケーラブルな推論が実現できることを示した
- 今後の課題
  - 潜在仮説空間の生成に CPI のアイデアを応用
    - [Riedel 06] の Markov Logic Networks における CPI のアナロジー
  - 言語処理タスクにおける評価 (c.f. 若手発表17: “杉浦ら, 談話関係認識への連想情報の応用”)