The Syntactic Process

9.2 Toward Psychologically Realistic Parsers
9.3 CCG Parsing for Practical Applications

(自然言語処理システム論 :7/11)

コンピュータ科学専攻 米澤研究室
M1　佐藤秀明

概要

- CKY Parser(Section 9.1.2) の改良
  - 事前にもつ知識の活用
- 実際のアプリケーションへの応用

Plausibility の導入

- 各 constituent のもっともらしさを計算
  - もっともらしさ = 現実にあり得る可能性
- あらかじめ用意した文脈との整合性を評価
  - 様相命題の notation を流用

(改良)CKY アルゴリズム

for j := 1 **to** n **do**

begin

t (j -1, j) := {A|A is a lexical category for a_j }

for i := j -2 **down to** 0 **do**

begin

t (i, j) :=

{A|there exists k, i<k<j, such that BC⇒A for some B ∈t (i, k), C ∈t (k, j), and not present(A, i, j)}

t (i, j) := rank (t (i, j))

end

end

rank 関数

- Constituents を plausibility の高い順に整列
- 先頭の constituent のみに plausibility '1' を付与
  - その他はすべて plausibility '0'
  - 別の戦略も可能だが、簡単のために採用しない

文脈

- 知識ベースを様相命題の集合で表す
- “◇” のついている event は :
  - 起こり得る、かつ
  - Accommodate(≒instanciate) できる

文脈の例

- (15)

$$person'x \wedge person'z \rightarrow \Diamond send'xyz$$
$$person'x \wedge person'y \rightarrow \Diamond summon'xy$$
$$\Diamond arrive'x$$
$$doctor'x \rightarrow person'x$$
$$patient'x \rightarrow person'x$$
$$flowers'x \rightarrow \neg person'x$$

- 変数は暗黙に全称化 (∀) されている

Parsing の例 (1)

- 文 :”The flowers sent for the patient arrived”
- the と flowers を shift して reduce すると :

(17)

a. $S/(S\backslash NP) : \lambda\, p.\, \iota\, x.\, flowers'\, x \wedge px$

b. $(S/(S\backslash NP))/(N\backslash N) : \lambda\, q.\, \lambda\, p.\, \iota\, x.\, (flowers'\, x \wedge qx) \wedge px$

Parsing の例 (2)

- “ι” は definite existential quantifier
  - ι で指定した部分が文脈に 1 つだけ存在することを要求
    - a. では” flowers'x”
    - b. では” flowers'x∧qx”
  - 文脈にはどちらも存在しない
    - plausibility をどう評価するか？

Parsing の例 (3)

- 解決策：条件の緩いほうに高い plausibility を付与
  - a. : ″flowers'x″ → plausibility '1'
  - b. : ″flowers'x∧qx″ → plausibility '0'
- a. のみについて accommodation(≒instantiation) を実施

  (18) $flowers'\,gensym'_1$
  - gensym'$_1$ は任意の定数
  - (18) を文脈に追加

Parsing の例 (4)

- 次に読む sent は 3 種類のカテゴリをもつ

(19)

a. $((S\backslash NP)/PP)/NP : \lambda x.\lambda y.\lambda z.\, send'\, yxz$

b. $(S\backslash NP)/PP : \lambda x.\lambda y.\, summon'\, xy$

c. $(N\backslash N)/PP : \lambda x.\lambda p.\lambda y.\, py \wedge send'\, yxsomeone'$

Parsing の例 (5)

- (19a,b) は (17a) とマッチする :

  (20)

  a. $(S/PP)/NP : \lambda y.\lambda z.\iota x.\ flowers'\ x \wedge send'\ zyx$

  b. $S/PP : \lambda y.\iota x.\ flowers'\ x \wedge summon'\ yx$

- (20a,b) は各々以下の命題を文脈に要請

  (22)

  a. $\Diamond\ flowers'\ x \wedge send'\ zyx$

  b. $\Diamond\ flowers'\ x \wedge summon'\ yx$

  – どちらも否定されている→低い plausibility を付与

Parsing の例 (6)

- (19c) は (17b) とマッチする :

  (21) $(S/(S\backslash NP))/PP : \lambda\, y.\, \lambda\, p.\, \iota\, x.$
  $(flowers'\, x \wedge send'\, yxsomeone') \wedge px$

- (21) は以下の命題を文脈に要請

  (23) $\Diamond\, flowers'\, y \wedge send'\, zyx$

  – 消去法で (21) に高い plausibility を付与

- (21) について accommodation を行う

  (24) $send'\, z\, gensym'_1\, someone'$

  – (24) を文脈に追加

    - Gensym'$_1$ は (18) で既出の定数

Parsing の例 (7)

- 次に for を shift して reduce
  - reduce できるのは (20b) と (21) のみ

  (25)

  a. $S/NP : \lambda y.\, \iota x.\, flowers' x \wedge summon' yx$

  b. $(S/(S\backslash NP))/NP : \lambda y.\, \lambda p.\, \iota x.$
  $(flowers' x \wedge send' yxsomeone') \wedge px$
- 文脈による Plausibility の評価
  - flower は summon の主語になれない→ a:'0'
  - flower は send の目的語になれる→ b:'1'

Parsing の例 (8)

- the patient の最も plausible なカテゴリは (26)

  (26) $NP : \lambda\, p.\, \iota\, x.\, patient'\, x \wedge px$

  - 条件の制約が最も緩い
  - 最も rank が高い
- (26) を accommodate

  (27) $patient'\, gensym'_2$

Parsing の例 (9)

- (26) は (25a,b) のどちらともマッチする

  (28)

  a. $S : \iota\, y.\ patient'\, y \wedge \iota\, x.\ flowers'\, x \wedge summon'\, yx$

  b. $S/(S\backslash NP) : \lambda\, p.\, \iota\, y.\ patient'\, y \wedge \iota\, x.$
  $(flowers'\, x \wedge send'\, yx someone') \wedge px$

- (28a) は再び implausible
- (28b) は plausible
  - Flowers を patient に send できる
  - Flowers と patient の実体がそれぞれ存在
  - $send'\, gensym'_2\, gensym'_1\, someone'$　が導ける

Parsing の例 (10)

- 最後に arrive を読んで終了
  - (28b) にマッチ
  - arrive の主語は何でもとれる

(改良)CKY Parser の特徴

- Plausibility に関わらず legal な constituent はすべて生成
- 曖昧な語の正しい解釈がそれを読んだ時点で判明
  - 例 :flower は send しない

他の Parsing 例 (1)

- “The doctor sent for the patient arrived” の場合
  - Modifier analysis より tensed-verb analysis が好ましい (?)
    - Doctor sent for the patient は plausible
    - Simple NP の前提条件が少ない

(30)

a. $S : \iota\, y.\, doctor'\, y \wedge \iota\, x.\, patient'\, x \wedge summon'\, xy$

b. $S/(S\backslash NP) : \lambda\, p.\, \iota\, y.\, doctor'\, y \wedge \iota\, x.\, (patient'\, x \wedge send'\, xysomeone') \wedge px$

他の Parsing 例 (2)

- すでに 1 人の医者が文脈に存在する場合

  (31) *doctor 'dexter '*

  - Null 文脈上での分析とほとんど同じ
  - Plausibility は低い (?)

- 2 人の医者が文脈に存在する場合

  (32) *doctor 'dexter ', doctor 'warren '*

  - (30a) は失敗
  - (30b) は高い plausibility を得る (?)

さらなる改良

- もっとやりたいこと
  - どの文と文脈が garden path を起こすのか
  - constituent の構築に関する優先順位を考慮したい
- より積極的なアルゴリズムが効果的
  - Beam-searching CCG Parser
  - Best-first chart-parser
  - plausibility のとりうる値を 0 と 1 の間の小数に
  - plausibility が閾値に達しない constituents を枝刈り

実用的なアプリケーション

- CKY は最悪 n^3 かかる
  - Vijay-Shanker and Weir のものは n^6
    - 複雑な構造共有がネック
- 実世界の文に対する平均計算量は許容範囲 [Komagata, 1997a, 1999]
  - 意味論的明瞭化や統計的最適化すら不要

CKY のバリエーション

- 様々な Parsing アルゴリズムに CCG を組み込める
  - 各々の文法理論に対し中立
  - 各々の文法が許す限り incremental なアルゴリズム
- 共通課題 :Plausibility の評価基準となる知識の構築が困難
  - 枝刈りできずに無駄な constituents が増加

Part-of-Speech-Tagging methods

- POS methods: 最初の単語の時点で枝刈り
  - 単語のもつカテゴリをよくあるものからｎ番目までに制限
  - Syntax とは独立に実施可能
- POS と CCG を組み合わせる
  - POS のカテゴリ種を CCG で拡張

Probabilistic Dependency Grammars (?)

- CFG の拡張
  - Rule に対して確率を算出
  - 最尤法や backing-off method(?) を使う
- 現在最も正確な Parser
  - Wall Street Journal に対し精度・再現率ともに 88%
- 非常に一般化された手法
  - Competence grammar 、Parsing アルゴリズム、確率の理論的整理と統合
- CCG は dependency を無制限な破片構造へ導入したところが面白い

展望

- Syntax から直接 plausibility を計算できるか？
  - 統計の蓄積は用いない
- より能動的な解釈 (推論など) が必要
  - Structural ambiguity
  - Attachment ambiguity