修士論文

生成語彙論に基づいた Abstract Meaning Representation Parsing

山元 勇輝
奈良先端科学技術大学院大学
先端科学技術研究科
情報理工学プログラム

主指導教員: 松本 裕治 教授
自然言語処理学研究室（情報科学領域）

令和 2 年 3 月 13 日提出

本論文は奈良先端科学技術大学院大学先端科学技術研究科に
修士 (工学) 授与の要件として提出した修士論文である。

山元 勇輝

審査委員：

松本 裕治 教授	（主指導教員，情報科学領域）
中村 哲 教授	（副指導教員，情報科学領域）
新保 仁 准教授	（副指導教員，情報科学領域）
進藤 裕之 准教授	（副指導教員，情報科学領域）

生成語彙論に基づいた Abstract Meaning Representation Parsing*

山元 勇輝

内容梗概

Abstract Meaning Representation (AMR) は, 自然言語文の意味を有向非巡回グラフで表現した意味表現形式である. AMR parsing の研究では入力文を正しい AMR に構造解析することが求められる. 既存研究の問題点として, 論理的多義のような動的な意味の生成を必要とする言語現象に対応できないことがあげられる. 本研究では生成語彙論で提唱されている生成的な演算を利用することで, これらの言語現象に対応した AMR の表現を提案し, 提案形式に出力できるよう既存の AMR parser を拡張する.

キーワード

AMR, 意味表現, 意味構造解析, 生成語彙論, 論理的多義

*奈良先端科学技術大学院大学 先端科学技術研究科 修士論文, 令和 2 年 3 月 13 日.

Abstract Meaning Representation Parsing based on Generative Lexicon*

Yuki Yamamoto

Abstract

Abstract Meaning Representation (AMR) is a meaning representation formalism which represents a natural language sentence as a directed acyclic graph. AMR parsing task is to parse input sentences into their corresponding AMRs. One of the problems in previous studies is that existing parsers cannot handle linguistic phenomena such as logical polysemy which require dynamic generation of non-compositional semantics. In this research, by adapting ideas from Generative Lexicon theory, we show expected output formats to represent these phenomena and propose extentions to an existing AMR parser.

Keywords:

AMR, meaning representation, semantic parsing, Generative Lexicon, logical polysemy

*Master's Thesis, Graduate School of Science and Technology, Nara Institute of Science and Technology, March 13, 2020.

Contents

List of Figures

List of Tables

1. 序論

本節では, 本研究の背景, 目的, 構成について述べる.

1.1 背景

計算機による自然言語の意味理解のためには, 自然言語を機械可読な形式に変換することが必要である. そのため, 意味表現や意味構造解析の研究においては, 自然言語の意味に対応する表現形式が提案され, 文をその形式に変換する取り組みがなされてきた. Abstract Meaning Representation (AMR)[3] も意味表現の一つであり, AMR 解析 (AMR parsing) の研究が近年盛んとなっている. 文から正確な AMR を獲得することができれば, 情報抽出や機械翻訳, 質問応答などの自然言語処理のアプリケーションで活用することができる. AMR parsing はこれらの応用先の基盤技術となるため, 解析精度の改善を追求することは重要な動機である. その一方で, 人間の創造的な言語使用に対応するためには, AMR で取り扱える表現の幅を広げることも不可欠である. 本研究のように既存の AMR や AMR parser に修正を加えることで対応できる表現の拡張を目指すアプローチである.

先行研究が対応できない表現に論理的多義がある. 例をあげると“The boy began the book.”という文は「本を始めた」という解釈ではなく“The boy began to read the book.”や“The boy began reading the book.”と同じ意味を表すことが可能である. このとき, 動詞 *begin* は自身の統語範疇を保持しつつ, 動詞句, 動名詞句と名詞句の選択が可能であるという点で論理的多義であるとみなされる. あるいは, “The girl baked a potato.”と “The girl baked a cake.”における動詞 *bake* は 2 つの関連した意味を表すものの, それぞれ「熱を加え続ける」と「熱を加える続けることで人工物を作る」という異なるタイプの事象を表すため, 論理的多義であるといわれている. AMR parsing の訓練の際は一般的に AMR コーパスを利用するが, 訓練データが論理的多義における意味解釈を考慮していないため, 望ましい出力結果を得ることができない. そのため, 解析アルゴリズムに工夫を加える必要がある.

1.2 目的

本研究では, これまで取り扱えなかった論理的多義の表現について, 先行研究の一つである CAMR[32] の遷移型アルゴリズムを拡張することにより, 提案する出力表現を獲得することを目指す. 近年, 文法理論の語彙化が進む中, **生成語彙論** [27]

では語彙に関する従来の静的な見方からの脱却を唱え, 論理的多義の解釈を動的に生成する生成的な演算を導入している. 特に, タイプ強制と共構成は動詞と目的語の選択において制約を緩和する役割を果たすものである. これらを解析アルゴリズムに組み込むことにより, 統語では明示されない生成的な解釈を AMR で表現する. その方法として, まずは生成的な演算を適用するべき表現を検出する. そして, 検出された表現に対して一連の生成的な演算を加えることにより新たな解釈を表現する. このようにして, AMR parser で取り扱える表現の幅を拡張する.

1.3 構成

本論文の構成は次の通りとなる. 2 節では, 自然言語処理及び理論言語学におけるこれまでの意味へのアプローチを概観し, 本研究の立場を明確にする. 3 節では, 生成語彙論の立場から, 既存研究における AMR parser の問題点について指摘し, 解決のために適切な意味表現を提案する. そして, 4 節ではその意味表現に解析するための手法を提案する. その後, 5 節で実験設定を示し, 6 節で実験結果と獲得した AMR について考察する. 最後に 7 節でまとめと今後の展望を述べる.

2. 関連研究

2.1 文の意味構造解析

これまでの自然言語処理における意味研究では, 語義曖昧性解消, 共参照解析, 意味役割付与など意味の部分的な側面を捉える意味解析の研究がなされてきた. そして, これらのタスクを支えるために様々な言語資源や評価手法が個別に構築されてきた. 例えば, PropBank[23] は動詞の述語項構造にのみ注目しているが, NomBank[21] では一般名詞や動詞の名詞化形など名詞に関する項構造を記述している. Banarescu ら [3] はこのように意味を細分化して個別に解決しようとする状態を “Balkanized” であると形容している. 一方, 統語の分野では階層構造に Part-of-Speech のアノテーションを加えた Penn Treebank[20] の構築により統語構造解析の性能が向上してきた. その結果, 単名詞句の特定など個別タスクを文の構造解析の副産物として取り扱うことが可能となっている. 意味においても同様の効果が期待され, 文の意味を意味表現に変換する意味構造解析の研究が取り組まれるようになった. そのような意味表現の中でもコーパスの規模が比較的大きいものに Discourse Representation Structure (DRS)[16] や AMR が含まれる.

2.2 Abstract Meaning Representation

2.2.1 AMR の特徴

AMR は文の意味をノードとエッジからなる有向非巡回グラフで表す意味表現である. ノードは文中の英単語, “walk-01” のような PropBank のフレーム, あるいは “date-entity” のようなキーワードのいずれかを指し, concept と呼ばれている. 一方, エッジは relation と対応しており, “:arg0”のような PropBank の意味役割や “:cause”, “:domain”, “:name”といった concept 間の意味関係を表している.

AMR を特徴付ける点として, 文を構成する要素の抽象化があげられる. 冠詞や前置詞のような機能語, 全称記号, そして時制や数のような語の屈折は省略されている. また, 統語的な複雑さが抽象化されている. 例えば, (1) において動詞 *fear*, 名詞 *fear*, 形容詞 *afraid* は同じ意味を表すため, 全て同じ concept である “fear-01” に置き換えられる. この場合, 文全体が表す意味も同じであるため, 図 1 のように同じ AMR 形式で記述される.

(1) a. The soldier was afraid of battle.

b. The soldier feared battle.

c. The soldier had a fear of battle.

```
(f / fear-01
    :ARG0 (s / soldier)
    :ARG1 (b / battle-01))
```

Figure 1 例文 (1) の AMR 形式

AMR グラフの構造は依存構造木に類似しているが, reentrancy を認めているという点で決定的に異なる. Reentrancy とは 1 つのノードを複数の親ノードの子として扱うことを指し, コントロール構文のように埋め込み節の主語が動詞の主語と同じである場合にみられる. AMR 形式で記述する際は, concept の ID を表す variable を用いて参照することになる.

このような特徴を含んだ AMR の例として, 文“ The boy wants to visit New York City.”に対応するグラフは図 2 のようになる. 文中の内容語である “boy”, “wants”, “visit”, “New York City”は, それぞれ “boy”, “want-01”, “visit-01”と 固有表現の “New York City”の concept として表現されている. また, “want-01”はコントロール動詞であるため, “boy”は “want-01”と “visit-01”の子ノードとして共有されている.

2.2.2 AMR Parsing

AMR コーパス [3] の構築が進むにつれて, 様々な AMR parser が提案されてきた. それらはアルゴリズムの違いによって, グラフベース, 遷移ベース, 文法ベース, seq2seq のカテゴリーに分類することができる. この節では, それぞれの代表的な研究を紹介する.

グラフ型には, 初めて提案された AMR parser の JAMR[10] がある. JAMR は concept identification と relation identification のパイプラインで構成されている. Concept identification のステージでは, 最もスコアの高いスパンとそれに対応する concept を系列ラベリングの問題として推測している. Relation identification では, maximum spanning connected subgraph (MSCG) というアルゴリズムを提案している. MSCG によって, 前のステージで予測した全ての concept を含み, 2 つのノード間に必ず 1 つのエッジを含み, かつ全てのノードがエッジを辿ることで他の全

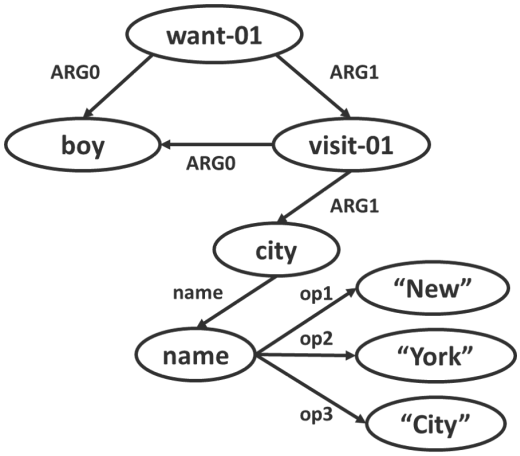

Figure 2 AMR グラフの例

てのノードに到達可能という 3 つの条件を満たしたグラフを予測している. 構文解析で利用される最大全域木アルゴリズムと同じように, 各ステップで最大スコアを持つエッジを予測することでグラフが完成する. Werling ら [34] は JAMR の concept identification を拡張している.

遷移型の CAMR では, 入力文を依存構造解析して得られた依存構造木に対し, 逐次的にアクションを加えていくことでグラフ構造に変換している. このように木構造をグラフ構造に変換する手法は特に tree-to-graph やツリー型と呼ばれており, 後続の Goodman ら [13] と Wang ら [31] によって取り組まれている. その他のアプローチとして, Damonte ら [7] が提案した AMR-Eager がある. 構文解析で利用される Arc-Eager アルゴリズム [22] を修正したものであり, 入力文を直接的にグラフ構造に変換する AMR parser である. AMR-Eager では AMR グラフの non-projective な特性に対処するため, stack の一番上のノードとその兄弟ノード間の reentrancy を復元できるようにアクションを拡張している. また, Ballesteros and Al-Onaizan[2] は stack-LSTM を利用したモデルを提案している.

文法ベースのアプローチには, 外部の意味資源を利用し文法推定を行った Artzi ら [1] や Peng ら [24] の研究がある. 前者は Combinatory Categorial Grammar (CCG) によるもので, 最初のステージで入力文を未完全な論理形式に変換し, 次のステージで情報の不足部分を決定するというアプローチがとられている. 後者は文法として Synchronous Hyperedge Replacement Grammar (SHRG) を利用している. 文を文脈自由文法で, AMR グラフを Hyperedge Replacement Grammar (HRG) でモデ

ル化し, それぞれをソース, ターゲットとして Synchronous Grammar を定式化している.

Seq2seq の AMR parser では, 問題となる学習データの少なさや前処理の方法に工夫が加えられている. Barzdin and Gosko[4] は初めて seq2seq AMR parser を提案し, 文字ベースの入力文を文字ベースで線形化された AMR に変換している. Konstas ら [18] は大量の教師なしデータを利用し, Paired Training という半教師あり学習を行っている. Zhang ら [35] は, Pointer-Generator Network を拡張することで reentrancy の解析精度を向上させ, 現時点での state-of-the-art を達成している. また, 統語構造や意味構造のような言語学的情報を前処理の段階で付与したり, モデル自体に含めようとする取り組みも存在している [11].

2.3 意味に関する理論

2.3.1 語彙意味論

Chomsky によって Lexicalist Hypothesis[6] が提唱されて以来, 語彙を統語から分離することで, それぞれを独立したモジュールとして取り扱う語彙主義の文法研究がなされてきた. この立場に基づいた文法理論である LFG[17] や HPSG[26] では, 統語と意味の情報を語彙項目に記述することで統語・意味構造の解析を行ってきた. また, 語彙主義に基づいた代表的モデルの一つに動詞クラス分類 [19] がある. このモデルでは, 意味の近い動詞群は統語的に同じ振る舞いをすると主張し, 同じ動詞クラスとしてまとめることができると述べている. 動詞クラス分類は WordNet[9] や VerbNet[29] へと拡張が進められてきた.

このように, 語彙主義の枠組みで語の意味と統語現象の対応関係が模索されてきた中で, 特に動詞を一義的に統語と対応づける観点からの脱却を図り, 新たな理論が提案されている. 構文文法論 [12] では, 非構成的な構文的意味を中心単位として扱うことで従来の構文と動詞の選択関係を見直している. 動的意味を導入した枠組みのうち, 生成語彙論 [27] では, 詳細な語彙構造を仮定することで語の創造的な多様性を説明している. また, 概念意味論 [15] では個別の語の構成を超えて, 概念構造と文脈の解釈を導入した意味の説明を試みている.

2.3.2 生成語彙論

本節では多義性の取り扱いについて整理して生成語彙論の立場を明確にした後, 多義性へのアプローチとしての生成的な演算について説明する.

多義性にはContrastive Polysemyと Complementary Polysemyの2通りが存在する [27][33]. 前者は, 一つの語が無関係な語義を保持している場合であり, 例として「銀行」や「土手」など複数の異なる語義を持つ名詞 bank があげられる. 後者は, それぞれの語義に関連がみられる多義性を指し,「羊」と「羊肉」を意味する名詞 lamb がそれに分類される.

Pustejovskyは多義性をモデル化した従来のアプローチを Sense Enumeration Lexicon (SEL) と呼び, 次のように定義している.

> A lexicon L is a Sense Enumeration Lexicon if and only if for every word w in L, having multiple senses $s_1, ..., s_n$ associated with that word, then:
>
> (i) if $s_1, ..., s_n$ are contrastive senses, the lexical entries expressing these senses are sorted as $w_{s_1}, ..., w_{s_n}$.
>
> (ii) if $s_1, ..., s_n$ are complementary senses, the lexical entry expressing these senses is stored as $w_{\{s1,...,sn\}}$.

SEL では語の異なる語義を列挙することで語彙辞書が構成される. その性質上, Contrastive Polysemy については適切に扱えると見なされている. しかし, Complementary Polysemy については, (2) における形容詞 *good* のように必ずしも説明できないという批判がある.

(2) a. Mary finally bought a good umbrella.
b. After two weeks on the road, John was looking for a good meal.
c. John is a good teacher.

(2a)-(2c) における *good* について, SEL の定義に従えば, $good_1$, $good_2$, $good_3$ の語義をそれぞれ “to function well”, “tasty”, “to perform some act well”として与えることができる. しかし, このアプローチでは語が新しい文脈で使用されるごとに語義を与えなければならず, *post-hoc* なものとなる.

生成語彙論においては, Complementary Polysemy の中でも統語範疇が保持されるものを特に **Logical Polysemy (論理的多義)** と呼んでいる. 論理的多義の性質を持つ語は SEL のように語彙辞書の中で複数の語義が列挙されるのではなく一つの中心的な語義を持つものとして扱われている [28]. そして, その語義から語彙的な意味の解釈と**生成的な演算**の適用により得られる生成的な意味の解釈が派生して生み出される考えられている.

$$\left[\begin{array}{l}\textbf{bank}_1\\ CAT = \textbf{count\_noun}\\ GENUS = \textbf{financial\_institution}\end{array}\right]$$

$$\left[\begin{array}{l}\textbf{bank}_2\\ CAT = \textbf{count\_noun}\\ GENUS = \textbf{shore}\end{array}\right]$$

Figure 3 Sense Enumeration Lexicon における Contrastive Polysemy の記述例

$$\left[\begin{array}{l}\textbf{lamb}\\ SENSE_1 = \left[\begin{array}{l}CAT = \textbf{mass\_noun}\\ GENUS = \textbf{meat}\end{array}\right]\\ SENSE_2 = \left[\begin{array}{l}CAT = \textbf{count\_noun}\\ GENUS = \textbf{animal}\end{array}\right]\end{array}\right]$$

Figure 4 Sense Enumeration Lexicon における Complementary Polysemy の記述例

HPSG などの制約に基づいた理論では, 制約に違反する表現は文法的に不適格であるとみなしてきた. しかし, 実世界では制約を違反していても問題なく使用される表現が存在する. そのような例として (3) がある.

(3) The boy began a book.

本来, *begin* は事象を表す目的語をとるという制約を持つた動詞であり, (3) では物体を表す *book* が目的語となっているため不適格とみなされるはずである. しかし, 実際は「少年が本を (読み/書き) 始めた」という解釈が可能な文である.

また, (4) も同様に解釈可能な例である.

(4) Mary baked a cake yesterday.

制約に基づけば *bake* は個体を目的語にとり,「熱を加え続ける」という change-of-state の意味で用いられる動詞である. しかし, (4) では食品が位置しており本来

ならば不適格な文であるが, *bake* について「熱を加え続けることでケーキを作り出した」という creation の解釈が可能である.

以上のように, 制約に違反しつつも容認される解釈を生み出す操作のことを生成的な演算と呼ぶ. 生成的な演算は制約を緩和し, より柔軟に意味を捉えるために定義されたものである. 動詞に関わる生成的な演算には**タイプ強制**と**共構成**が存在する. タイプ強制は引数のタイプが関数の要求するものと異なる場合、引数のタイプを合うように強制的に変更する操作である. (3) では *begin* が関数となり, *book* が引数となる. 共構成は引数が関数の要求するタイプのものと異なる場合に、引数と関数の両方の変更を試みて、両者のタイプが合うように変更する操作である. (4) では *bake* が関数となり, *cake* が引数となる.

(3), (4) では生成的な演算を通して新しい解釈が生み出された. その結果, *begin* は目的語位置に to 不定詞や動詞の進行形をとるだけでなく 名詞の目的語をとることが可能となった。 また, *bake* は従来の change-of-state に加えて creation という新しい意味を持つようになった. このことから, *begin* と *bake* は論理的多義の性質を持つ動詞であるとみなすことができる.

3. 問題点

前節では AMR parsing の先行研究を概観したが, いずれの提案手法も論理的多義の現象を正しく取り扱うことができない. 先行研究の一つである CAMR で解析した場合も, 反映されるべき意味の解釈を AMR で表現することができない. ここでは前節で使用した例文 (3), (4) について, CAMR による解析結果と生成的な演算が生み出す解釈を比較し, その問題点を指摘する.

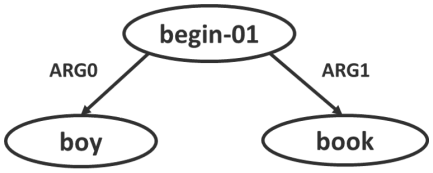

Figure 5 “The boy began a book.” の解析例

$$\begin{bmatrix} \textbf{begin} \\ \text{EVENTSTR} = \begin{bmatrix} \text{E1} = \textbf{transition} \\ \text{E2} = \textbf{transition} \\ \text{RESTR} = <\circ_{\propto} \end{bmatrix} \\ \text{ARGSTR} = \begin{bmatrix} \text{ARG1} = \textbf{x}\text{: }\textbf{human} \\ \text{ARG2} = \mathbf{e_2} \end{bmatrix} \\ \text{QUALIA} = \begin{bmatrix} \text{FORMAL} = \mathbf{P}(\mathbf{e_2}, \mathbf{x}) \\ \text{AGENTIVE} = \textbf{begin\_act}(\mathbf{e_1}, \mathbf{x}, \mathbf{e_2}) \end{bmatrix} \end{bmatrix} \quad \begin{bmatrix} \textbf{book} \\ \text{ARGSTR} = \begin{bmatrix} \textbf{x}\text{: }\textbf{info} \\ \textbf{y}\text{: }\textbf{physobj} \end{bmatrix} \\ \text{QUALIA} = \begin{bmatrix} \text{info·physobj\_lcp} \\ \text{FORMAL} = \text{hold(y,x)} \\ \text{TELIC} = \text{read(e,w,x.y)} \\ \text{AGENTIVE} = \text{write(e',v,x.y)} \end{bmatrix} \end{bmatrix}$$

Figure 6 *begin* と *book* の語彙構造

まずは (3) の解析結果に着目する. 図 5 をみると, “begin-01” の PropBank の記述にしたがって ARG0 に行為者である “boy” を, ARG1 に行為の内容である “book” をとっており, 「少年が本を始めた」という AMR グラフが出力されている. 次に生成的な演算による生成の流れを確認すると *begin* の項構造 (ARGSTR) には第 2 項に e_2 (事象) を要求するという制約の記述がある. しかし, *book* は事象ではなく physobj (個体) である. このように, *book* は *begin* に要求されているタイプではなく不整合が生じる. そこでタイプ強制により, *book* の特質構造 (QUALIA) からその意味と関連した行為の解釈が導き出され, *read* か *write* が補完される. (3) の場合, 文脈的判断から *read* が補完されることで, 「少年が本を読み始めた」という解釈が可能となる. 図 5 の解析結果ではこのように補完された事象を明示的に表すことができていないため不適切である.

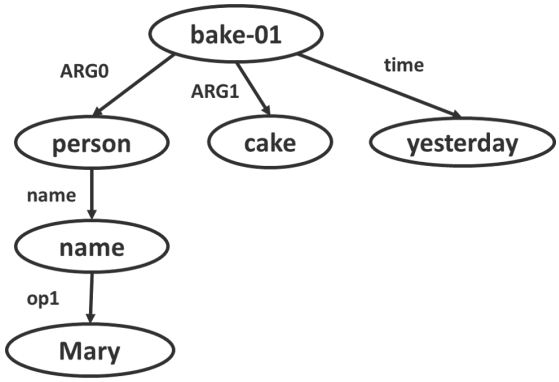

Figure 7 “Mary baked a cake yesterday.” の解析例

Figure 8 *bake* と *cake* の語彙構造

CAMR で (4) を解析した場合, AMR は図 7 のように表現される. このグラフにおける *bake* は, change-of-state の意味として解析されている. そのため, ARG0 に行為者を, ARG1 に「熱を加える対象」をとっており,「ケーキに熱を加え続けた」という解釈になる. 項構造における制約の記述をみると, 本来 *bake* は *potato* のように physobj (個体) の意味タイプをもつ語を要求するはずである. しかし, food (食品) である *cake* をとっているため不整合が生じている. この不整合を解消するために, *cake* が *bake* の情報を読み取り, 互いの特質構造を一致させる. これにより Product の性質をもつ目的語に対して, *bake* に creation の意味が生成される. 図 7 は ARG1 の *cake* を「*bake* という行為によって作られた食品」としか表すことができず, *potato* などと同様に「*bake* という行為を受け続ける個体」として表現しているので AMR として適切ではない.

以上の内容をまとめると, CAMR では PropBank のフレームが一義的に利用されてしまっているため, 生成された解釈を表現することができない. AMR コーパスに出現するほとんどの述語に PropBank のフレームに対応するものが存在す

るといわれており, 解析の際にそのフレームが語彙的な制約としてはたらいている [3]. そのため, 新しい解釈を表すためにはこのフレームをもとに, 動的な生成に関わる語彙情報の要素を組み込む必要がある.

4. 提案手法

4.1 出力表現の提案

3 節の問題点に対する指摘を受け, 生成的な演算を通して生み出される解釈を適切に表現した AMR を提案する. (3) における *begin* の述語項構造とタイプ強制による事象の補完は図 9 のように表されている. この図では目的語を選択する *begin* が *book* を事象として扱い, その情報を名詞句内に埋め込んでいる. ここで, *begin* はコントロール動詞であるため *read* は *boy* を主語にとることに注意すると, この木構造に対応する AMR グラフは図 10 のように表すことができる. グレー箇所が前節の図 5 との差分であり, 生成的な演算によって導出された新しい意味を表現する部分となっている. また, 特質構造に記述された *read* を “read-01” というノードとして明示化することにより, (5) と等価な解釈を AMR で表現している.

次に, (4) に共構成が適用された AMR を図 11 のように表す. 2 つのノード “bake-01” と “cake” の間に新たなノードとエッジを追加することで, 語彙的意味である change-of-state の場合とは異なる動作を表現している. *Bake* の生成的意味は creation であり, これは素材から *cake* を作るまでの動作のことを指す. そこで, 生成されたノードに “exist-01” のラベルを付与することで, 目的語 *cake* が「*bake* の結果作り出されたもの」であるという意味を持たせている.

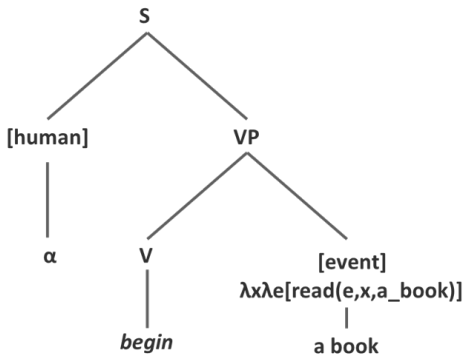

Figure 9 動詞 *begin* による事象の補完

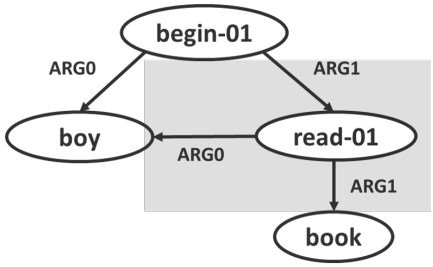

Figure 10 Type Coercion を適用した AMR グラフの例

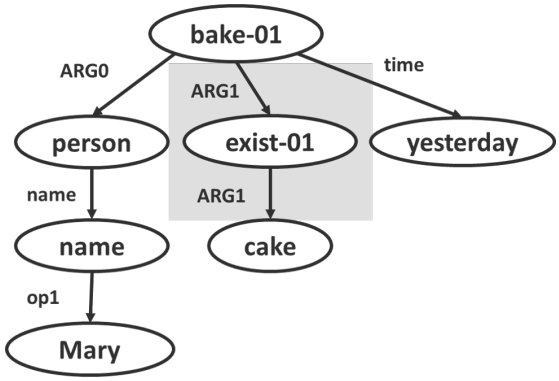

Figure 11 共構成を適用した AMR グラフの例

4.2 遷移型 AMR Parser

本研究では Wang らで提案されている CAMR の遷移型アルゴリズムの実装[1]を利用する. CAMR は入力文を依存構造解析[2]にかけて依存構造木 D を得てから AMR グラフに変形するツリー型の AMR parser である. CAMR では AMR コーパスの Gold データをもとに, グラフの状態に対して最適なアクションを推測する分類器を学習する. 前段階として依存構造木のノードを帰りがけ順でバッファに整列し, バッファの先頭要素にアクションを適用していくことで concept や relation の推測が行われていく. CAMR の内部状態は $S = (\sigma, \beta, G)$ と定義され, 次のように表される.

[1]https://github.com/c-amr/camr

[2]https://github.com/BLLIP/bllip-parser と https://github.com/stanfordnlp/CoreNLP

- σ: 現在まで未処理のノードを格納するバッファである. σ の先頭を σ_0 とし, $\sigma = \sigma_0|\ \sigma'$ と表す.

- β: σ_0 の子ノードを格納するバッファである. β の先頭を β_0 とし, $\beta = \beta_0|\ \beta'$ と表す.

- G: 現在の依存構造木の状態を表す. 依存構造木 D で初期化されている.

依存構造木に加えられるアクションは以下の 8 通り存在する. l_r と l_c はそれぞれ relation と concept の推測を伴うアクションであることを示している.

- NEXT-EDGE-l_r

  現在のエッジ (σ_0, β_0) に relation ラベル r を付与し, β から先頭要素 β_0 を取り除く. "The police wants to arrest ..." という文において "wants" が σ_0 で "police" が β_0 のとき, それらの間のエッジに "ARG0" を付与する.

- SWAP-l_r

  σ_0 と β_0 の親子関係を入れ替える. そして, エッジ (β_0, σ_0) に relation ラベル r を付与する. 例えば, "South Korea and Israel oppose..."における "Korea" が σ_0 で "and"が子ノードの先頭要素 β_0 であるとき, "and"が "Korea"の親ノードとなりノード間のエッジに "op1"が付与される.

- REATTACH$_k$-l_r

  エッジ (σ_0, β_0) を取り除き, β_0 をその他のノード k に接続する. そして, 新しく作られたエッジ (k, β_0) に relation ラベル r を付与する. "South Korea and Israel oppose..."の "Korea"が σ_0 で "Israel"が子ノードの先頭要素 β_0 であるとき, 並列構造を作るために "Israel" が "Korea" から切り離されてその他のノードである "and" と接続し, "and" と "Israel" 間のエッジに "op2" が付与される.

- REPLACE-HEAD

  σ_0 に接続するエッジを取り除き, β_0 と置き換える. そして, β_0 を σ の先頭要素とする. さらに, β に子ノードを展開する. 例えば, "live in Singapore" において "in" が σ_0 のとき, AMR に不要であるので接続するエッジが取り除かれ, β_0 である "Singapore" に置き換えられる.

- REENTRANCEk-l_r

  あるノード k と β_0 の間にエッジ (k, β_0) を加えて relation ラベル r を付与する. “The police wants to arrest …” という文において “want” が σ_0 で “police” が β_0 のとき, “want’ がコントロール動詞であることから “police” を主語にとるノード k (この場合, “arrest”) を探し, k と β_0 を接続してラベルを付与する.

- MERGE

  σ_0 と β_0 を取り除き, 新たに作られた $\tilde{\sigma}$ を σ の先頭要素とする. “Michael Karras” という人名があり, σ_0 と β_0 がそれぞれ “Michael” と “Karras” であるとき, “Michael Karras” という新しいノード $\tilde{\sigma}$ にまとめる.

- NEXT-NODE-l_c

  β が空の場合のみ実行可能である. σ_0 に対して concept ラベル c を付与する. そして, σ から先頭要素 σ_0 を取り除き, 新たに σ の先頭要素となった σ_1 の子ノードで β を初期化する. “The police wants to arrest …” において “police” が σ_0 で “wants” が σ_1 のとき, σ_0 に concept が付与されてバッファから取り除かれる. そして “wants” の子ノードで β が初期化され, 現在 σ_1 の “wants” が次の σ_0 となる.

- DELETE-NODE

  β が空の場合のみ実行可能である. σ_0 をグラフ G と σ から取り除く. “The police wants to arrest …” において “The” が σ_0 のとき, AMR には不要なのでグラフとバッファから取り除かれる.

4.3 生成的な演算の実装

CAMR の解析では, まず分類器が予測する最適なアクション系列が求められ, 入力文の依存構造木に適用されていく. 本研究では解析途中のグラフの状態を参照し, 不整合が特定された場合に生成的な演算を適用する.

4.3.1 不整合の検出

本節ではまず, 不整合な状態について説明する. タイプ強制における不整合とは, アスペクト動詞 σ_0 の子 β_0 がいずれも事象を表さない場合である. そこで, 依存

事象	イメージ	説明	動詞の例
State	− − −	定常の状態	*be, love, know*
Process	～～～	線的な過程	*run, push, drag*
Transition	×	点的な動作	*give, open, build*

Table 1 生成語彙論の事象タイプ分類

構造解析の結果を利用し, エッジ (σ_0, β_0) が “dobj” である場合にタイプ強制を適用する. 次に共構成における不整合の求め方を説明する. 前節でみてきたように, 動詞の意味解釈の差はそれが表す事象の違いに起因するものである. これに関連して, 生成語彙論ではアスペクト情報に基づいた動詞分類 [30] に倣い, State, Process, Transition の 3 つの事象が定義されている. 事象の分類に従えば, change-of-state が表す事象は Process であり, creation は Transition となる. これをイメージ化すれば, 語彙的意味の change-of-state は線的な過程である一方, 生成された意味である creation は点的な動作として出発点と目的点をもつことになる.

共構成が行われるためにはこれらの点に相当する情報が必要である. そこで, 本研究では新たな解釈を生み出すための語彙情報として, 対象の動詞に 2 つのスロットと head を導入する. 各スロットには動詞がとり得る目的語が対応しており, head が付与されたスロットは動詞が要求する目的語として典型的なものを示す. *build* クラスの動詞に関しては出発点は Material であり, 目的点は Product と考えることができるため, head は Material に対応したスロットに付与される. このような head の語彙情報に加えて, Material や Product の性質を表す名詞のクラスタを獲得することにより, 対象とする動詞に対して目的語が不適切か否かを判断する.

4.3.2 ノード生成のアクション

不整合が特定された場合, AMR parser の状態 $S = (\sigma, \beta, G)$ に基づいて, 生成的な演算として定義した一連のアクションを加える. これらのアクション系列は TYPECOERCION, COCOMPOSITION としてそれぞれ表 2, 3 に記述した通りである. 生成的な演算を適用する目的は文に現れない意味的要素を明示することにあるので, 新たにノードを生成するアクションとして GENERATE-l_c[3]を導入する.

[3]CAMR においても固有表現を表すために’person’ や’name’ といったノードの生成自体は行われている. 一方, GENERATE-l_c はノードの生成に加え, 内容語のラベルを推測して付与するアクションである. これを利用することで, 文中に明示されない concept を表すことが可能となる.

- GENERATE-l_c σ_0 と σ_1 の間のエッジを取り除く. その後, 新たにノード x を生成し, concept ラベルを付与する.

タイプ強制においては GENERATE-l_c で初めにノードを生成し, 明示されていない事象を表す concept を言語モデルを用いて推測する. 後続のアクションで, 現在のノードに関する concept と relation を付与する. そして, 生成したノードとその主語にあたるノード k の間に REENTRANCEk-l_r によってエッジを加え, relation を付与する. 共構成においても GENERATE-l_c により初めにノードを生成するが, concept ラベルには動詞の語彙に事前に記述された情報 (e.g. “exist-01”) を付与する.

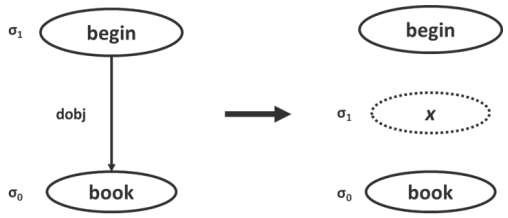

Figure 12 GENERATE-l_c の例

アクション	現在の状態	次の状態	ラベル
GENERATE-l_c	$(\sigma_0 \mid \sigma_1 \mid \sigma', [\,], G)$	$(x \mid \sigma_1 \mid \sigma', \beta', G')$	$\gamma\ [x \rightarrow l_c]$
NEXT-NODE-l_c	$(\sigma_0 \mid \sigma_1 \mid \sigma', [\,], G)$	$(\sigma_1 \mid \sigma', \beta = CH(\sigma_1, G'), G')$	$\gamma\ [\sigma_0 \rightarrow l_c]$
NEXT-EDGE-l_r	$(\sigma_0 \mid \sigma', \beta_0 \mid \beta', G)$	$(\sigma_0 \mid \sigma', \beta', G')$	$\delta\ [(\sigma_0, \beta_0) \rightarrow l_r]$
REENTRANCE	$(\sigma_0 \mid \sigma', \beta_0 \mid \beta', G)$	$(\sigma_0 \mid \sigma', \beta_0 \mid \beta', G')$	$\delta\ [(\beta_0, k) \rightarrow l_r]$
NEXT-NODE	$(\sigma_0 \mid \sigma_1 \mid \sigma', [\,], G)$	$(\sigma_1 \mid \sigma', \beta = CH(\sigma_1, G'), G')$	NONE

Table 2 TYPECOERCION のアクション系列

アクション	現在の状態	次の状態	ラベル
GENERATE-l_c	$(\sigma_0 \mid \sigma_1 \mid \sigma', [\,], G)$	$(x \mid \sigma_1 \mid \sigma', \beta', G')$	$\gamma\ [x \rightarrow l_c]$
NEXT-NODE-l_c	$(\sigma_0 \mid \sigma_1 \mid \sigma', [\,], G)$	$(\sigma_1 \mid \sigma', \beta = CH(\sigma_1, G'), G')$	$\gamma\ [\sigma_0 \rightarrow l_c]$
NEXT-EDGE-l_r	$(\sigma_0 \mid \sigma', \beta_0 \mid \beta', G)$	$(\sigma_0 \mid \sigma', \beta', G')$	$\delta\ [(\sigma_0, \beta_0) \rightarrow l_r]$
NEXT-NODE	$(\sigma_0 \mid \sigma_1 \mid \sigma', [\,], G)$	$(\sigma_1 \mid \sigma', \beta = CH(\sigma_1, G'), G')$	NONE

Table 3 COCOMPOSITION のアクション系列

5. 実験準備

5.1 CAMR の学習

実験では Linguistic Data Consortium (LDC) が公開している AMR コーパスをデータセットとして利用し, CAMR の実装をもとに分類器の学習を行う. AMR コーパスには, 新聞記事やインターネット掲示板などの文とそのアノテーションのペアが含まれている. しかし, 文と AMR のアラインメントは記述されていないため, JAMR Aligner[10] を利用する. また, CAMR は LDC2017T10 で頻出する複文やハイパーリンクなどを取り扱うことができない. このことを考慮し, 今回の実験では AMR Annotation Release 1.0 (LDC2014T12)[4]を利用する. LDC2014T12 の割り当ては以下の表の通りである.

Train	Dev	Test	Total
10,312	1,368	1,371	13,051

Table 4 LDC2014T12 の割り当て

5.2 生成的な演算のための準備

5.2.1 言語モデルの利用

タイプ強制により生成されたノードのラベルを推測するために, 学習済みの Bert-Base Uncased[8] を利用する. Bert-Base Uncased は Masked LM をタスクとして学習された Transformer の双方向の言語モデルである. Masked LM による学習では文中のランダムな単語を “[MASK]” に置き換え, その部分に相当するトークンを正確に推測できるようにモデルを訓練している. 本研究では, アスペクト動詞と目的語の間に “to [MASK]” を追加することで, 文中で明示されない事象を言語モデルに推測させる.

5.2.2 クラスタの作成

共構成のための不整合を特定するためには 4.3 節で提案したスロットに含まれるクラスタが必要である. そこで, Material か Product を表す名詞のクラスタを獲得

[4]https://catalog.ldc.upenn.edu/LDC2014T12

するため, 今回は $k = 2$ の K-means アルゴリズムを利用する. Levin の動詞分類 [19] に従い, Material/Product Alternation を起こす動詞クラスから動詞 *arrange, assemble, blow, build, carve, cast, compile, cut, develop, embroider, fold, grow, hammer, hatch, knit, mold, pound, roll, sew, shape, spin, stitch, weave* を選択し, Wikipedia Corpus[5]から共起する直接目的語の学習データを獲得する. 素性には学習済みの GloVe[25] の分散表現を用いる. また, 距離関数には単純なユークリッド距離 $d(\mathbf{x}_1, \mathbf{x}_2) = \sqrt{\sum_i^n (x_{1i} - x_{2i})^2}$ を利用する.

5.3 AMR parser の評価

5.3.1 評価データの作成

LDC2014T12 には本研究で解析の対象とする表現は含まれていない. そこで, Wikipedia Corpus からタイプ強制と共構成が適用されるような文を抽出し, 評価データをそれぞれ作成する. タイプ強制は動詞がアスペクト動詞に分類されコントロール動詞として用いられる場合に生じる現象である [28]. 今回は条件を満たす動詞のうち *begin, start, end, finish* を対象とする. また, 共構成を適用する動詞群として [27] で取り上げられている動詞のうち *build* クラスに分類される *bake* 動詞 *bake, cook, fry* を選択する. なお, データ作成にあたっては公開されている AMR-Editor[14] を使用した.

生成的な演算	含まれる動詞	Total
タイプ強制	*begin, start, end, finish*	35
共構成	*bake, cook, fry*	20

Table 5 作成した評価データの内訳

5.3.2 評価方法

AMR parsing の評価方法として Smatch スコア [5] が存在する. Smatch は与えられた 2 つの AMR グラフの類似度を測る尺度である. AMR parser の精度を測定するために, 評価データの解析結果と Gold の AMR 間に適用する.

[5]https://dumps.wikimedia.org/enwiki/latest/

```
(a / want-01
    :ARG0 (b / boy)
    :ARG1 (c / go-01
        :ARG0 b))
```

Figure 13 “The boy wants to go.”の AMR グラフ

```
(x / want-01
    :ARG0 (y / boy)
    :ARG1 (z / football))
```

Figure 14 “The boy wants the football.”の AMR グラフ

まず, AMR グラフが relation, variable, concept の 3 つ組に分解される. “The boy wants to go.”と“The boy wants the football.”の AMR は図 13, 14 に表されている通りである. これら 2 つの Smatch を計算するために, 図 15, 16 のように変換する.

```
instance(a, want-01) ^
instance(b, boy) ^
instance(c, go-01) ^
ARG0(a, b) ^
ARG1(a, c) ^
ARG0(c, b)
```

Figure 15 “The boy wants to go.”の 3 つ組

次に 2 グラフ間の variable の全組み合わせについて Precision, Recall, F_1 スコアを計算する. 例えば, $x = c, y = b, z = a$ の場合はそれぞれ 2/5, 2/6, 0.36 となる. 全ての場合について計算すると, $x = a, y = b, z = c$ のときの F_1 スコアが 0.73 となり最大になる. このとき, 2 グラフの Smatch スコアは 0.73 となる.

それぞれの AMR グラフの variable 数を n, m とした場合, $n!/(n-m)!$ 通りの F_1 スコアを計算することになるが, variable の数が多い場合, 多大な計算時間を要するため現実的でない. そこで, 整数線形計画法を用いて最適解を求める問題に定式化している. これは NP 完全であり, 最適解を多項式時間で計算できるアルゴリズムは存在しない. しかし, 多くの場合, 貪欲な山登り法を利用することで求められ

```
instance(x, want-01) ^
instance(y, boy) ^
instance(z, football) ^
ARG0(x, y) ^
ARG1(x, z)
```

Figure 16 “The boy wants the football.”の 3 つ組

るため, [5] ではこの手法で実装されている.

6. 実験結果

6.1 評価

Parser	Precision	Recall	F_1
w/o GL	0.63	0.69	0.66
w/ GL	0.63	0.69	0.66

Table 6 LDC2014T12 における生成的な演算の有無による評価

まず, 生成的な演算の追加による解析全般の影響を調査するため, LDC2014T12 で用意されている Test データを対象に解析を行い Smatch スコアを計算した. 結果は表 6 の通りである. w/o GL は CAMR のみの性能であり, w/ GL は CAMR に生成的な演算を追加した場合の性能を指している. 表 6 によると生成的な演算の有無に関わらず F_1 が 0.66 となっている. また, Test データの解析において不必要に GENERATE-l_c のアクションが選択された場面は確認できていない. 以上から, 提案手法による拡張によって AMR parser 自体の性能は損なわれていないことが示唆される.

生成的な演算	Precision	Recall	F_1
TC	0.75	0.79	0.77
CC	0.67	0.69	0.68
TC_{Gold}	0.78	0.81	0.79
CC_{Gold}	0.67	0.70	0.69

Table 7 作成データにおける生成的な演算の評価

次に, 不整合の検出と生成的な演算の適用が正確に実行されているのかを調査するため, 作成した評価データと解析結果のペアの類似度を Smatch スコアにより計算し, 表 7 に示した. 評価データのうち TC はアスペクト動詞を含みタイプ強制が適用される文を解析した場合である. 一方, CC は *bake* 動詞のいずれかを含んだ文を対象に解析したものである. 実験の結果, TC と CC の F_1 はそれぞれ 0.77 と 0.68 となった. 表 6 より CAMR の性能を示す w/o GL の F_1 が 0.66 であるということも考慮すれば, 目標となる表現に概ね解析できているといえる.

比較のため, 生成的な演算として正解の動作を適用した場合の Smatch スコアを計算した. TC_{Gold} はタイプ強制をすべて検出し, 生成された事象のノードにすべて正解ラベルを付与できた場合の評価である. CC_{Gold} は動詞の語彙情報をもとに共構成の適用をすべて正しく判断できた場合の評価である. 結果をみると TC_{Gold} は 0.79 となり, TC と比較して 0.02 の差がみられた. また CC_{Gold} は 0.69 となり, CC との差は 0.01 となった. 以上のことから, 数値上わずかではあるが適切に適用されていない動作が存在することが明らかになった. 解析ミスが不整合の検出と生成的な演算による動作のいずれに起因するのかを調べる必要がある.

6.2 エラー分析

本節では実験結果を受け, 解析ミスが生じた要因を追究する. そのために, 生成的な演算に関連した問題のある解析例を取り上げて分析する.

```
(x3 / cook
        :ARG0 (x1 / they)
        :ARG1 (x6 / or
                :op2 (x7 / vegetable)
                :op1 (xap0 / exist-01
                        :ARG1 (x5 / meat)))
        :location (x11 / water
                :mod (x10 / boil-01)))
```

Figure 17 “They would cook the meat or vegetables in the boiling water.”の解析結果

Bake 動詞の解析結果に最も頻繁に表れたミスとして, concept ラベルの付与があげられる. この種類のミスでは図 17 における *cook* のように, 入力文のトークンのままの形 (e.g. *cook, cooked, cooking*) で表現されている. 解析結果では図 17 の例だけでなく *bake* や *fry* の場合もそのまま concept として表されていた. このようなミスは relation の推測にも影響が及ぶことがあるので問題である. 要因として考えられるのは Train データの少なさである. PropBank には動詞のフレームが数多く記述されているが, AMR コーパス中に出現するものはそのうちの一握りであるため, 学習結果から推測することは困難であると考えられる, 改善のためには, concept の推測の際に PropBank など動詞の情報を利用することが必要である.

また, 共構成の検出においてもミスが生じていた. 図 17 の *cook* は *meat* と *vegetable* という 2 つの目的語を持ち, 片方の *meat* は “exist-01” と接続されている. *Bake* の head スロットは Material であり *meat* は Material に分類されるため, 4.2 節の手法に従えばここでの *bake* は change-of-state の語彙的意味となるはずである. しかし, クラスタの分類の段階でミスが生じてしまっているため, 不必要に共構成が適用されている. 評価データでは 20 例中 16 例正解であり, 残り 4 例の目的語 *meat* と *bird* は Product に分類されていた. このことから, クラスタリングの素性として生物性/無生物性の属性を取り入れることが有効であると考えられる.

```
(x8 / begin-01
        :time (x2 / date-entity
                :year 1931)
        :time (x5 / airship)
        :ARG0 (x6 / person
                :name (n / name
                        :op1 "Graf"
                        :op2 "Zeppelin"))
        :ARG1 (xap0 / start-01
                :ARG0 x2
                :ARG1 (x12 / service
                        :frequency (x9 / regular)
                        :ARG1-of (x10 / schedule-01)
                        :mod (x11 / passenger)
                        :prep-against (x15 / and
                                :op1 (x14 / country
                                        :name (n1 / name
                                                :op1 "Germany"))
                                :op2 (x16 / continent
                                        :name (n2 / name
                                                :op1 "South"
                                                :op2 "America"))))))
```

Figure 18 “In 1931, the airship Graf Zeppelin began regular scheduled passenger service between Germany and South America.”の解析結果

タイプ強制の場合, 不整合の検出にはミスが生じていなかった. その一方で, 生成した concept のラベルの推測においてミスが見られた. 図 18 では *begin service*

で明示されない事象として本来は “offer-01” であるが “start-01” を付与している. このように解析結果と正解ラベルが完全に一致しなかったのは 35 例中 25 例であり, 改善が求められる. 推測されたラベル, 正解ラベル, 目的語を (推測, 正解, 目的語) の 3 つ組で表すと, 一致しない例には (“write-01”, “read-01”, *draft*), (“show-01”, “exhibit-01”, *work*), (“film-01”, “direct-01”, *film*) のような種類のものが存在した. 1 例目のラベルは特質構造における主体役割[6]と目的役割[7]に対応するものであり, *draft* に関する 2 つの側面を表しているため, 文脈次第では “write-01” が正解となるような例である. 2 例目は concept が表す内容が類似している例である. これらの例に関しては, ほとんどの場合言語モデルで推測する際の 5 つの候補に正解が含まれていることを確認したので, 適切に絞り込む手法を考える必要がある, 3 例目も類似した concept の対であるが, 推測ラベルは文脈中に出現する語の影響を強く受けている例である. この例における *film a film* のような表現は同族目的語構文と呼ばれるものであり, 同じ内容の語が連続して使用されるものである. 容認される語は少ない (e.g. *dream, fight, etc...*) ため, 絞り込みの際にそれらの動詞以外には選択制限を課して, 文中に出現する語と同じラベルを推測させないことが考えられる. さらにこの制限を動詞クラスに拡大すると, 図 18 のように “begin-01” と “start-01” が連続することを防ぐことができると考えられる.

[6]主体役割は物体の発生に関する要因を表す. (e.g. *book* の主体役割は *write*)

[7]目的役割は物体の目的や機能を表す. (e.g. *book* の主体役割は *read*)

7. 結論

7.1 まとめ

本研究の成果は以下の 2 点である.

- 論理的多義に対応する AMR の表現方法を提案した.
- 生成的な演算を AMR parsing で実現した.

本研究では生成語彙論の観点から CAMR のアルゴリズムを拡張し, AMR において意味の動的な生成を実現するための手法を提案した. 特に, タイプ強制と共構成を実現するためにアスペクト動詞と *bake* 動詞を対象に論理的多義の現象に取り組み, 統語上表現されないが意味的に存在する要素の表現が可能になった. 一方, Smatch スコアによる評価とエラー分析を通して, 共構成による意味の判別やタイプ強制で生成されたノードのラベル付与に関して改善の余地があることがわかった.

7.2 展望

今後の改善点として, 共構成を適用する判断をより正確にするために, クラスタリングの素性として生物性の有無を考慮する必要がある. また, タイプ強制における concept ラベルの推測で, 候補を絞り込むためにいくつかの条件を規定する必要がある. 生成語彙論に基づいた拡張の方向性としては, 今回指定した特定の動詞に限定せず生成的な演算を実装し, 統合的な AMR parser の構築を目指したい. 例えば, 副詞や形容詞を組み込んで選択束縛を実装することが考えられる. さらに, 動詞によるタイプ強制のみならず, 使役動作構文のように構文単位でのタイプ強制についても AMR で表現することが考えられる. また, “He buttered a toast.”と “He built a wooden house.” における「塗られた素材」と「使われた素材」で対比されるように, 必ずしもその役割が明示されない要素の扱いについても考える必要がある. この枠組みで提案されている default/shadow argument に関する情報を考慮して, 明示されないノードを必要に応じて補完することも今後の課題とする.

謝辞

まず初めに, 松本裕治教授に感謝します. 人文系出身の私を自然言語処理学研究室に受け入れてくださり, 研究で進捗が出ない時期も徹底して助言・指導を賜りました. 心より感謝します.

また, 次の先生方に感謝します. 進藤裕之助教授には Parsing 勉強会で面倒を見ていただきました. 新保仁准教授には計算機の利用に関して助けていただきました. 中村哲教授には中間発表で有益なコメントをいただきました.

そして, 研究室の皆様に感謝します. 秘書の北川さんが研究室の環境を整えてくださったおかげで快適に研究活動を行えました. 先輩, 同期, 後輩の皆様とは研究内容は違えど各々の観点から意見を出し合えたことは貴重な経験でした. 特に先輩方には研究の立場について多大な理解をいただいたこと, 進路の相談に乗っていいただいたことに感謝します.

入学前にもお世話になった以下の方々に感謝します. 首都大学東京の小町守准教授にはご多忙の中, 当分野の研究の作法を丁寧に教えていただきました. また, 吉川さんと井上さんには入試の際にお世話になりました.

最後に, 精神的に支えてくれた家族に深く感謝します.

参考文献

[1] Yoav Artzi, Kenton Lee, and Luke Zettlemoyer. Broad-coverage ccg semantic parsing with amr. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1699–1710, 2015.

[2] Miguel Ballesteros and Yaser Al-Onaizan. AMR parsing using stack-LSTMs. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1269–1275, Copenhagen, Denmark, September 2017. Association for Computational Linguistics.

[3] Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. Abstract meaning representation for sembanking. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pages 178–186, 2013.

[4] Guntis Barzdins and Didzis Gosko. RIGA at SemEval-2016 task 8: Impact of Smatch extensions and character-level neural translation on AMR parsing accuracy. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 1143–1147, San Diego, California, June 2016. Association for Computational Linguistics.

[5] Shu Cai and Kevin Knight. Smatch: an evaluation metric for semantic feature structures. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 748–752, Sofia, Bulgaria, August 2013. Association for Computational Linguistics.

[6] Noam Chomsky. Remarks on nominalization. In Roderick A. Jacobs and Peter S. Rosenbaum, editors, *Readings in English Transformational Grammar*, pages 184–221. Ginn, Boston, 1970.

[7] Marco Damonte, Shay B. Cohen, and Giorgio Satta. An incremental parser for abstract meaning representation. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 536–546, Valencia, Spain, April 2017. Association for Computational Linguistics.

[8] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[9] Christiane Fellbaum. *WordNet: An Electronic Lexical Database*. Bradford Books, 1998.

[10] Jeffrey Flanigan, Sam Thomson, Jaime Carbonell, Chris Dyer, and Noah A Smith. A discriminative graph-based parser for the abstract meaning representation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1426–1436, 2014.

[11] DongLai Ge, Junhui Li, Muhua Zhu, and Shoushan Li. Modeling source syntax and semantics for neural amr parsing. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pages 4975–4981. International Joint Conferences on Artificial Intelligence Organization, 7 2019.

[12] Adele E. Goldberg. *Constructions: A Construction Grammar Approach to Argument Structure*. Cognitive Theory of Language and Culture Series. University of Chicago Press, 1995.

[13] James Goodman, Andreas Vlachos, and Jason Naradowsky. UCL+Sheffield at SemEval-2016 task 8: Imitation learning for AMR parsing with an alpha-bound. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 1167–1172, San Diego, California, June 2016. Association for Computational Linguistics.

[14] Ulf Hermjakob. Amr editor: A tool to build abstract meaning representations. In *manuscript*, 2013.

[15] Ray S. Jackendoff. *Foundations of Language: Brain, Meaning, Grammar, Evolution*. OUP Oxford, 2002.

[16] Hans Kamp and Uwe Reyle. From logic to discourse, 1993.

[17] Ronald Kaplan and Joan Bresnan. Lexical-functional grammar: A formal system for grammatical representation. *Lexical-functional grammar: A formal system for grammatical representation*, pages 173–281, 01 1982.

[18] Ioannis Konstas, Srinivasan Iyer, Mark Yatskar, Yejin Choi, and Luke Zettlemoyer. Neural AMR: Sequence-to-sequence models for parsing and generation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 146–157, Vancouver, Canada, July 2017. Association for Computational Linguistics.

[19] Beth Levin. *English verb classes and alternations : a preliminary investigation*. University of Chicago Press, 1993.

[20] Mitchell Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. Building a large annotated corpus of english: The penn treebank. 1993.

[21] Adam Meyers, Ruth Reeves, Catherine Macleod, Rachel Szekely, Veronika Zielinska, Brian Young, and Ralph Grishman. The nombank project: An interim report. In *Proceedings of the workshop frontiers in corpus annotation at hlt-naacl 2004*, pages 24–31, 2004.

[22] Joakim Nivre, Johan Hall, Jens Nilsson, Atanas Chanev, Gülşen Eryigit, Sandra Kübler, Svetoslav Marinov, and Erwin Marsi. Maltparser: A language-independent system for data-driven dependency parsing. *Natural Language Engineering*, 13(2):95–135, 2007.

[23] Martha Palmer, Daniel Gildea, and Paul Kingsbury. The proposition bank: An annotated corpus of semantic roles. *Computational linguistics*, 31(1):71–106, 2005.

[24] Xiaochang Peng, Linfeng Song, and Daniel Gildea. A synchronous hyperedge replacement grammar based approach for amr parsing. In *Proceedings of the Nineteenth Conference on Computational Natural Language Learning*, pages 32–41, 2015.

[25] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

[26] Carl Pollard and Ivan A. Sag. *Head-Driven Phrase Structure Grammar*. The University of Chicago Press, Chicago, 1994.

[27] James Pustejovsky. *The Generative Lexicon*. MIT Press, Cambridge, MA, 1995.

[28] James Pustejovsky and Pierrette Bouillon. Aspectual Coercion and Logical Polysemy. *Journal of Semantics*, 12(2):133–162, 04 1995.

[29] Karin-Kipper Schuler. *VerbNet: A Broad-Coverage, Comprehensive Verb Lexicon*. PhD thesis, University of Pennsylvania, 2006.

[30] Zeno Vendler. *Linguistics in Philosophy*. Cornell University Press, 1967.

[31] Chuan Wang, Sameer Pradhan, Xiaoman Pan, Heng Ji, and Nianwen Xue. CAMR at SemEval-2016 task 8: An extended transition-based AMR parser. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 1173–1178, San Diego, California, June 2016. Association for Computational Linguistics.

[32] Chuan Wang, Nianwen Xue, and Sameer Pradhan. A transition-based algorithm for amr parsing. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 366–375, 2015.

[33] Uriel Weinreich and William Labov. *Explorations in Semantic Theory*. Janua Linguarum. Series Minor. De Gruyter, 2014.

[34] Keenon Werling, Gabor Angeli, and Christopher D. Manning. Robust subgraph generation improves abstract meaning representation parsing. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 982–991, Beijing, China, July 2015. Association for Computational Linguistics.

[35] Sheng Zhang, Xutai Ma, Kevin Duh, and Benjamin Van Durme. AMR parsing as sequence-to-graph transduction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 80–94, Florence, Italy, July 2019. Association for Computational Linguistics.

発表文献

山元勇輝, 松本裕治. 語彙意味論に基づいた Abstract Meaning Representation Parsing. 言語処理学会第 25 回年次大会発表論文集, 名古屋大学, pp.505-506, March 14, 2019.