機械翻訳エンジン jaw について

玉置健二　角田慶太　松本忠博　池田尚志

岐阜大学工学部

1　はじめに

我々は、日本語から多言語への機械翻訳を行うパターン変換型の機械翻訳エンジン jaw を開発し、これを用いてこれまでに中国語、ミャンマー語、シンハラ語、ベトナム語、および日本手話を目的言語とする翻訳システムの構築を試みてきた。

機械翻訳エンジン jaw は、機械翻訳システムを構築するツールとして研究室のホームページ上で近日中に公開する予定である [1]。本稿では、jaw を用いた翻訳システム構築の方法について、サンプルとして構築した日英翻訳システム jaw/English を例に述べる。

2　機械翻訳エンジン jaw を用いた翻訳の流れ

jaw は VC++で作成された、日本語の構文構造を翻訳対象の言語の表現構造に置き換えて翻訳を行なうパターン変換方式かつトランスファー方式の機械翻訳エンジンである。解析、照合、生成、線状化の４つの処理で目的言語への翻訳を行う (図 1)。以下にそれぞれの処理の概要を示す。

2.1　解析と InputTree

日本語入力文の解析には、我々の研究室で開発中の日本語文節構造解析システム ibukiC[2] を用いている。ibukiC を通じて文節情報と各文節がどの文節に係るかを示す構文情報を得、それを元に日本語入力文の構文木構造 InputTree(IT) を作成する。

2.2　照合と TransferTree

作成された IT は日本語パターン翻訳規則辞書との照合を経て TransferTree(TT) に変換される。日本語パターンとは日本語の表現の中から目的言語との対応を取るのにふさわしい文の構造を部分的に取り出して一般形として表したものであり、それぞれに目的言語への翻訳規則が対応付けられている。

TT は日本語パターンの木構造 PatternTree(PT) の組み合わせとして作成される翻訳のための木構造であり、IT のデータと IT がどの様なパターンの集合であるかのデータを保持する。

2.3　生成と ExpressionTree

TT の部分木を構成する各日本語パターンに対応付けられた各翻訳規則をルートノードから再帰的に呼び

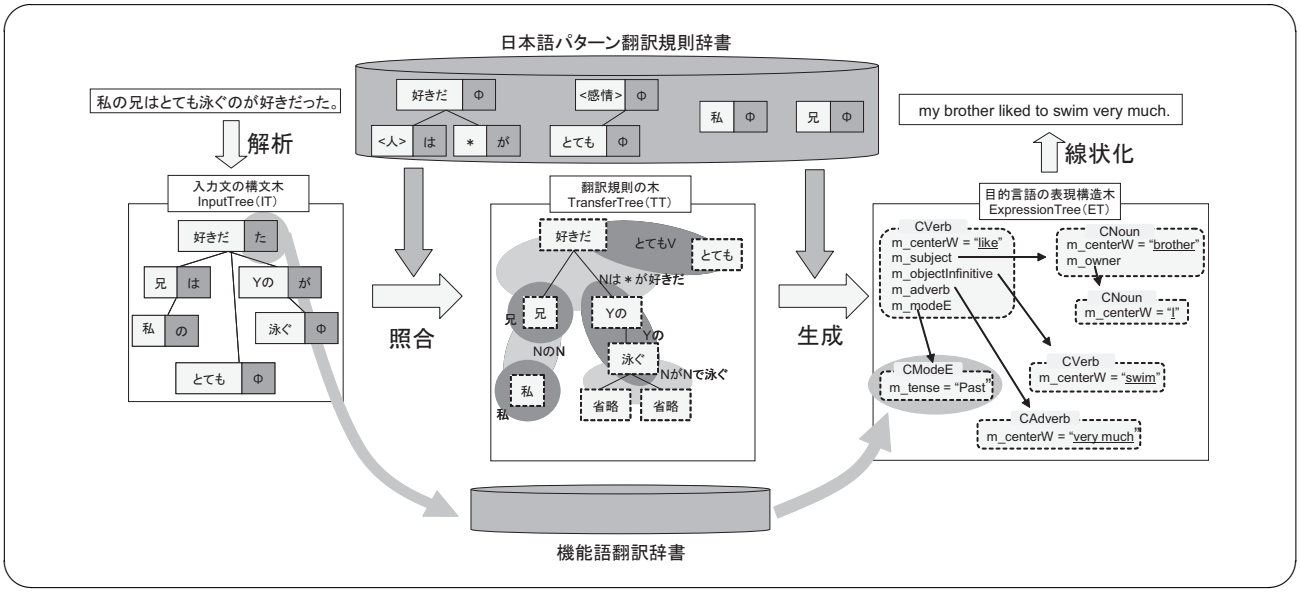

図 1: jaw の翻訳の流れ (jaw/English)

出し、目的言語の構造を表現した木構造に置き換える。この目的言語の表現構造木を ExpressionTree(ET) と呼ぶ。

ET は VC++のオブジェクトとして実現しており、各翻訳規則には、このオブジェクトを作成する C++の関数が対応付けられている。この関数群は、日本語パターン翻訳規則辞書に対し変換モジュールを実行することで自動的に翻訳規則の DLL としてまとめられる。

2.4 線状化による訳文の生成

ET の各オブジェクトのメンバーには、目的言語の言語表現を生成するための言語表現上の部品や必要な情報が記述されている。それらを参照しながら、あらかじめ定義しておいた目的言語の文法規則に従って語順調整や語形変化、冠詞や助動詞の付加などを行い翻訳文を作成する。また、ibuki の解析によって得られている機能語情報を使って機能語の翻訳に必要な処理も行う。この線状化処理は VC++のオブジェクトのメンバー関数として実現している。

3 jaw を用いた翻訳システムの構築

jaw は目的言語ごとに翻訳辞書をリレーショナルデータベース (RDB) 上で管理している。線状化関数を含むオブジェクトの定義は、目的言語ごとに VC++のプログラム上で行う。このようにして目的言語に依存する処理を翻訳エンジンから切り離し、新たな言語にも容易に対応できるシステムとなっている。jaw を用いた翻訳システムの構築は、以下の４つの作業を行うことで実現できる。

1. 目的言語のクラスの設計
   jaw は目的言語の表現構造木 ET を VC++のオブジェクトとして実現しているため、目的言語の言語構造を反映したクラスの設計を行う必要がある。

2. 線状化関数の実装
   各クラスのメンバーを部品として一次元の言語表現（文や句）を得るための線状化関数を実装する。これはクラスのメンバー関数として実現しており、語順や語形変化など、目的言語の言語構造を反映する。

3. 日本語パターン翻訳規則辞書の蓄積
   日本語の表現パターンとそれに対応した目的言語の翻訳規則を辞書に登録する。翻訳規則は表現構造木 ET を生成する規則であり、変換モジュールを用いて DLL に実装される。

4. 機能語翻訳辞書の蓄積
   機能語の翻訳は、ET 生成後に機能語翻訳規則辞書を参照し、ET を構成する各オブジェクトに必要な表現要素を書き込むことで行う。辞書に登録の無い機能語は翻訳文には反映されない。

3.1 目的言語のクラス設計

表現構造木 ET を構成する各ノードは目的言語の表現要素を保持する VC++のクラスのオブジェクトであり、クラスが持つ各メンバーによって、そのノードが表す目的言語上での言語表現のための部品や必要な情報が保持される。

基本的にクラスの作成は、同じメンバーを持ち、かつ同じ規則で線状化可能な表現要素を基準とする。多くの場合、品詞等の文法カテゴリーに対応することになるが、それにとらわれる必要は無く、システムを構築しようとする人の考え方に依存する。例として jaw/English に設けたクラスの種類と継承関係を図 2 に示す。

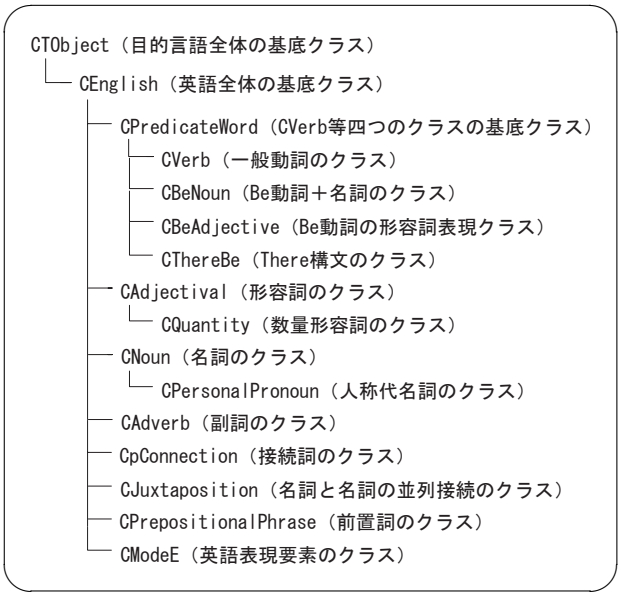

図 2: jaw/English のクラス一覧

CTObject クラスは jaw の各目的言語に共通する基底クラスであり、日本語の機能語情報や、オブジェクト自身を根とする部分表現構造を線状化して得られた目的言語テキストをメンバーに保持する。

CTObject クラスを継承する CEnglish クラスは英語共通の基底クラスであり、オブジェクト自身が英語のどの成分に相当するか（主語、目的語等）といった情報を保持する。

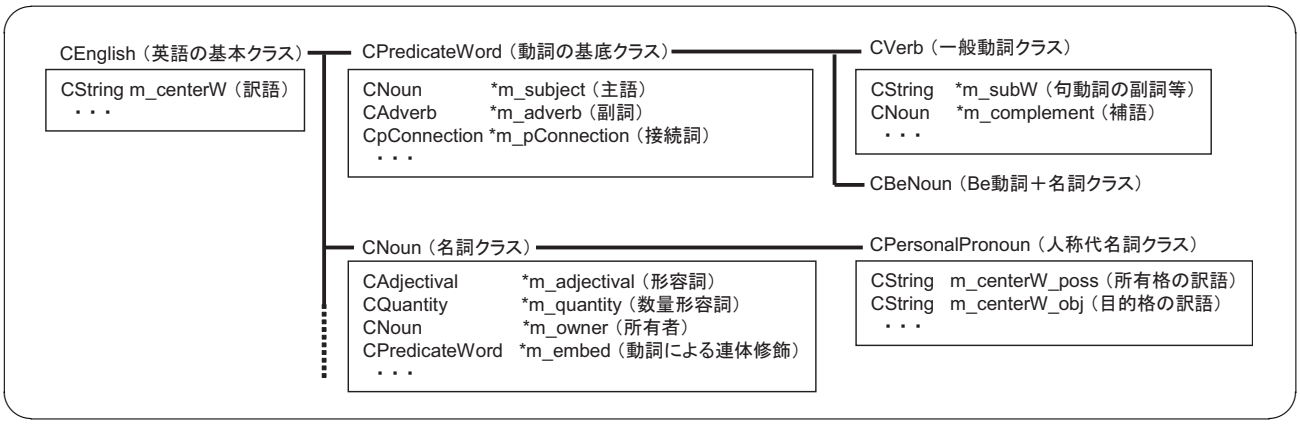

図 3: jaw/English の各クラスのメンバーの例

そこから派生するクラスには主語や目的語に使用される名詞クラス (CNoun クラス)、名詞の修飾を行う形容詞クラス (CAdjectival クラス) 等がある。各クラスの持つメンバーの例を (図 3) に示す。

クラスの定義は C++のソースコード中に直接埋め込んでいる。

3.2 線状化関数の実装

線状化 (linearize) 関数は、表現構造木 ET から訳文を作成する関数であり、設計した各クラスのメンバー関数として実装する。jaw/English の場合、一般的な ”SV”、”SVOO”などの文を作る CVerb クラス (一般動詞クラス) の線状化関数には、メンバーである ”m_subject”(主語) や ”m_dObject”(直接目的語) が指すオブジェクトの線状化関数を英語の語順に従った順番で呼び出す処理、不規則動詞の活用を行うために活用データベースを参照する処理などを実装しており、CNoun クラス (名詞クラス) の線状化関数には、名詞を修飾する ”m_adjectival”(形容詞) などの線状化関数を呼び出す処理、名詞の種類によって冠詞を付与する処理などを実装している (図 4)。

線状化関数の実装には目的言語の文法構造の知識が不可欠となる。

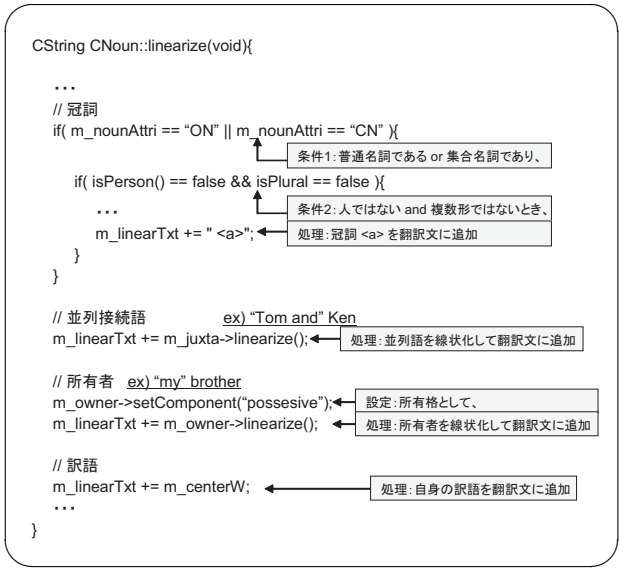

図 4: 名詞クラスの線状化関数の一部

3.3 日本語パターン翻訳規則辞書の蓄積

日本語パターン翻訳規則辞書は、日本語の表現パターンと目的言語の表現構造の対を集めたものであり、Microsoft Access の RDB 上に表現されている。

3.3.1 日本語パターン

日本語パターンとは日本語の表現の中から文の構造を部分的に取り出し、一般形として表したものである。1 つの日本語パターンは、そのパターンのキーとなる文節中の内容語あるいは機能語 (キーワード)、キーワード文節を修飾する文節とキーワード文節が修飾する文節に対する制約条件、受身使役系への読み替え規則といったパターンに対するその他のデータなどの情報から構成されている。

各文節における制約条件は同一のキーワードから成る表現をさらに細かく分類し、異なる日本語表現パターンとすることで、より精度の高い翻訳を実現する役割を果たしている。jaw では自立語条件として意味属性と字面を、機能語条件として任意機能語を指定することが可能である。また機能語制約は格助詞と格助詞相当語だけでなく文節中の任意の機能語を制約条件とすることができる。

日本語パターン翻訳規則辞書の内容は翻訳の精度に直結するもので、任意の入力文に対して正しい翻訳を行うためには、それに対応する数多くの日本語パター

ンが必要となる。jaw/English では、日英翻訳のための日本語パターンを大系化している日本語語彙大系 [3] のパターンの中からサンプルして利用した。

3.3.2 翻訳規則

翻訳規則とは日本語の表現パターンを目的言語の表現構造に変換するための変換規則である。「キーワードの訳語はどんな言葉になるか？」「各文節は目的言語上でどんな役割を果たすか？」といった翻訳の情報、実際にプログラム上で使用される際のフラグ変数の値などの情報などで構成され、RDB で管理されている。

日本語パターンと翻訳規則の例を図 5 に示す。

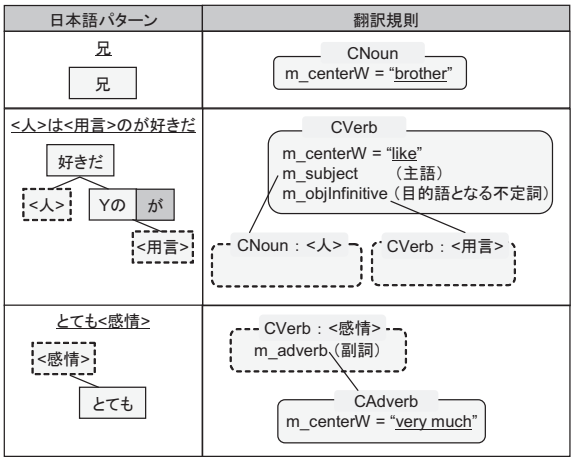

図 5: 日本語パターンと翻訳規則

3.3.3 日本語パターン翻訳規則辞書記述・編集システム jawEditor

日本語パターンやそれに 1 対 1 で対応する翻訳規則を直接テーブルに記述していくのは大変な作業である。そこで、より簡単にパターンや翻訳規則の記述、検索、後編集が行えるシステム jawEditor が用意されている。jawEditor は Microsoft Access のフォームで作成されている。

3.4 機能語翻訳辞書の蓄積

jaw では命題的内容を翻訳するための日本語パターン翻訳規則辞書とは別に、機能語を翻訳するための機能語翻訳辞書を持っている。機能語の翻訳は、ET が作成された後、機能語翻訳規則辞書を参照し、ET を構成する各オブジェクトに必要な表現要素を書き込むことで行う。

機能語翻訳規則辞書のテーブルには、日本語・目的言語間での対応が 1 対 1 になる場合に用いるもの (タイプ A) と、1 対多になる場合に用いるもの (タイプ B) との 2 種類がある。タイプ A には適合時の処理内容を、タイプ B にはそれに加え、表現構造中の要素の値を用いた訳し分け条件を記述する (図 6)。

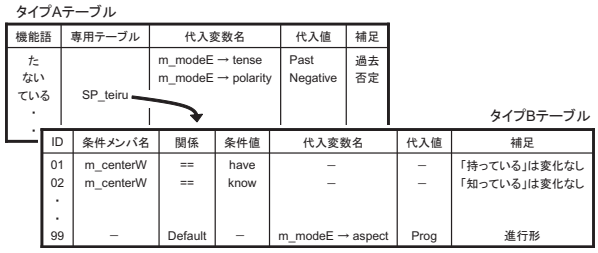

図 6: 機能語翻訳規則テーブル

4 jaw/English での翻訳例

jaw/English を用いた翻訳結果の例を以下に示す。

(1) もし明日雨なら、私は釣りに行かないだろう。	
正訳	If it rains tomorrow, I will not go fishing.
jaw	If it rain<s> tomorrow, I will not go fishing.

(2) ケンは 2 年間日本にいる。	
正訳	Ken has been in Japan for two years.
jaw	Ken has been at Japan for two year<s>.

(1) の形式主語 ”it” は日本語パターン「雨だ」および「雨が降る」の翻訳規則に主語 (”it”) の情報を記述することで出力している。

(2) では完了形の表現は正解しているが、場所を表す前置詞の判別処理が未実装なため、”at” という誤った出力となっている。

三単現の”s”や複数形名詞の”s”については、それを確定するアルゴリズムが未実装であるため、人間に判断を委ねるという意味で”< >”で囲って出力している。

5 おわりに

日本語から多言語への機械翻訳エンジン jaw の処理方式と、jaw を用いた翻訳システムの構築法について、英語への翻訳システムを例として述べた。

なお、日本語の解析システムは ibukiC である必要はないが、機能語の翻訳方法などシステムを若干変更する必要がある。

参考文献

[1] 池田研究室ホームページ
http://www.ikd.info.gifu-u.ac.jp/

[2] 機能文節を導入した文節構造解析システム ibukiC(v0.20) について：池田尚志 他：言語処理学会第 14 回年次大会 (2008)

[3] 日本語語彙大系：NTT コミュニケーション科学研究所 監修、池原悟 他 編集：岩波書店 (1997, 1999)