日本語から多言語への機械翻訳エンジン jaw

宇野修一　福本真哉　田中伸明　松本忠博　池田尚志

岐阜大学工学部

1　はじめに

我々はパターン変換型機械翻訳エンジン jaw の開発を行っている。jaw は日本語から多言語への機械翻訳を行うエンジンで、現在は中国語、ミャンマー語、シンハラ語、ベトナム語、および日本の手話を目的言語とする翻訳システムの構築を試みている。本報告では目的言語の表現構造への変換処理を中心に jaw システムの処理内容を述べる。

2　jaw の概要

jaw は現在盛んに研究されている統計的な手法を用いた手法ではなく古典的なトランスファー方式の翻訳システムである (図 1)。

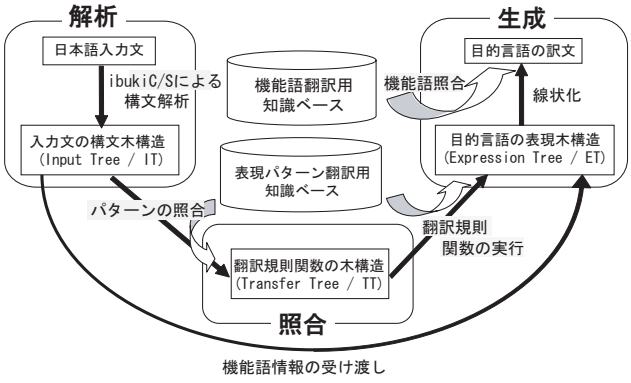

図 1: システムの概要

以下は jaw の特徴である。

- 翻訳規則を「命題的内容の翻訳用」「時制・話者の判断などの用言後接機能語の翻訳用」「体言後接機能語 (副助詞) などの取立て表現の翻訳用」の 3 種類に分けている。
- 翻訳規則から目的言語の構造を表現するオブジェクトの木構造を作り出すための C++の関数 (翻訳規則関数) を作成し、DLL として管理する。
- 目的言語の表現構造の各オブジェクトが目的言語の 1 次元言語表現を生成するための線状化関数 (生成関数) というメソッドを持つ。線状化関数をルートノードから再帰的に呼び出すことで訳文が得られる。

jaw は日本語から任意の言語への機械翻訳システムを構築するためのエンジンである。個々の目的言語への機械翻訳システムを構築するために必要なことは「目的言語の表現構造を表すクラスの設計」「各クラスにおける線状化関数の作成」「目的言語に対する翻訳規則の蓄積」の 3 つである。

3　表現構造への変換

入力された日本語文に対して図 2 に示すような文節構造解析・係り受け解析が行われ、入力文の木構造が作成される。本節では入力文の木構造 InputTree(IT) と表現パターン翻訳用知識ベースとの照合、および表現構造への変換処理について述べる。

```
これこそが{ 文節番号:1／係り先文節番号:3／
            品詞:名詞／CW:これ／
            FW1:こそ／FW2:が／FW3:Φ／
            FW4:Φ／FW5:連用／FW6:Φ　　}
決め手　{ 文節番号:2／係り先文節番号:3／
            品詞:名詞／CW:決め手／
            FW1:Φ／FW2:Φ／FW3:Φ／
            FW4:Φ／FW5:ダ系／FW6:Φ　　}
だ。　{ 文節番号:2／係り先文節番号:0／
            品詞:ダ系／CW:だ／
            FW1:Φ／FW2:だ／FW3:Φ／
            FW4:Φ／FW5:文末／FW6:。　　}
```

図 2: 文節構造と係り受け関係

3.1　表現パターン翻訳用知識ベース

表現パターン翻訳用知識ベースには命題的内容を翻訳するための翻訳規則として、文節間の関係を基に表現される日本語の表現パターンとそれに対応する翻訳規則が格納されている。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) を中心として記述する。キーワード文節が修飾する文節、キーワード文節を修飾する文節に対しては内容語部分に意味属性と字面を制約条件として指定することができる。また機能語部分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。表現パターンは次の 3 タイプに分類される。

・Base タイプ
「内容語をキーワードとし、どのような文節がその内容語を修飾するか」を記述するためのパターン。通常、用言の結合価構造や格構造として扱われるものと同じであるが、用言に限らず任意の体言に対しても作成される。Addition タイプのパターンは Base タイプのパターンを係り先文節 (親ノード) とする形で記述される。

図 3 は用言後接機能語を指定して作成したパターンの例である。jaw では図 3 のようなパターンを作成すると、通常「させる」をトリガーとして発生する使役の機能語に対する処理を省略するため、日本語では使役表現となるが目的言語では使役表現とならない表現への対処を容易に行うことができる。

・入力「Aが Bを 失望させる。」
・パターン「<体言>が <人>を 失望する＋させる」
→「A disappint B.」

図 3: 機能語条件も指定したパターンの例

・AdditionCW タイプ
「内容語をキーワードとし、その内容語を含む文節がどのような文節に係るか」を記述するためのパターン。「とても美しい」の「とても」のように自立語で係り先文節に情報を付加する役割を持つ副詞などが該当する。

・AdditionFW タイプ
「機能語をキーワードとし、その機能語を含む文節がどのような文節に係るか」を記述するためのパターン。「私の兄」「歌って踊る」の「の」や「て」のように文節同士をつなぎ合わせる役割を持つ機能語が該当する。

現在対応しているのは上記のような 1 つの単語 (キーワード) を中心とした、いわば局所的な表現パターンのみであるが、今後は文全体にかかわる、より大域的なパターンの取り扱いについても検討していく予定である。

3.2 日本語表現パターンとの照合

日本語表現パターンはリレーショナルデータベース (RDB) 上に記述されている。日本語表現パターンとの照合は (1)RDB 上でのキーワードの検索 (2) 検索されたパターンの木構造 PatternTree(PT) の作成 (3) 作成された PT の自立語条件・機能語条件と入力文の木構造 IT との照合 という処理を再帰的に繰り返すことによって行われる。

照合の結果、複数のパターンとマッチした場合には最適解の選択が行われる。最適解としては以下の 3 種の条件を総合し、コストが最も低いパターンを選択する。選択の原則は「一般的な規則より個別的、具体的な規則を優先するべきである」という考えである。

・木構造の構成に使用されているパターンの種類と数
パターンの種類としては Addition タイプよりも Base タイプを優先し、パターンの数がより少ない方を優先する。

・自動変形の有無などのパターンの使用状況
パターン変形などの特殊処理を行って照合に成功したパターンよりも登録されているパターンをそのまま使用して照合に成功したパターンを優先する。特殊処理にはそれぞれコストを付与しており、頻繁に使用する処理はコストを低く設定してある。

・意味属性間の距離
意味属性条件に関してはシソーラス上の意味属性間の距離が短い方を優先する。

3.3 表現パターン翻訳規則と表現構造の作成

翻訳規則は日本語表現パターンを目的言語の表現構造に変換するための規則である。日本語表現パターンとは 1 対 1 で対応付けられ、表現パターン翻訳用知識ベース編集システムである jawEditor を用いて RDB 上に記述される。個々の翻訳規則は翻訳規則関数への変換モジュールによって C++の関数に変換される。翻訳規則関数の関数名は対応する日本語表現パターンの RDB 上での主キーの番号である。作成された翻訳規則関数はコンパイルされ、それぞれの目的言語専用の DLL となる。翻訳規則の変換モジュールは変換した C++の関数をテキストに出力するため、jawEditor が未対応の特殊な翻訳規則に関してもその段階で人手で修正を加えることが可能である。

翻訳規則関数を実行すると目的言語の表現構造がオブジェクトのネットワークとして作り上げられる。現在、翻訳規則関数としては次の 4 種類を設けている。

・Base タイプ翻訳規則関数
指定されたクラスのインスタンスを作成し、そのインスタンスの指定されたメンバに引数として渡された子ノードのオブジェクトをセットする (図 4-(1))。Base タイプの表現パターンに対する翻訳規則関数として使用される。

・AdditionCW タイプ翻訳規則関数

指定されたクラスのインスタンスを作成し、そのインスタンスを引数として渡された親オブジェクトの指定されたメンバにセットする (図 4-(2))。AdditionCW タイプの表現パターンに対する翻訳規則関数として使用される。

・AdditionFW タイプ翻訳規則関数１

引数として親オブジェクトと子オブジェクトを渡され、親オブジェクトの指定されたメンバに子オブジェクトをセットする (図 4-(3))。AdditionFW タイプの表現パターンに対する翻訳規則関数として使用され、接続用のインスタンスを介さずに子ノードと親ノードを直接接続する。

・AdditionFW タイプ翻訳規則関数２

指定されたクラスのインスタンスを作成し、そのインスタンスの指定されたメンバに、引数で渡された子オブジェクトをセットする。また、引数で渡された親オブジェクトの指定されたメンバに、作成したインスタンスをセットする。つまり引数で渡された 2 つのオブジェクトを接続用に作成したインスタンスを中間において接続する (図 4-(4))。接続詞のような接続用のインスタンスの作成を必要とする AdditionFW タイプの表現パターンの翻訳規則関数として使用される。

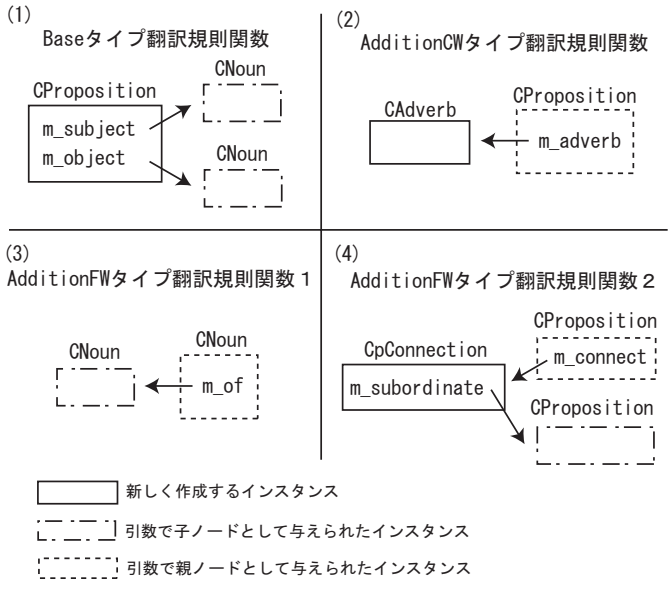

図 4: 表現構造の作成

これらの翻訳規則関数は、jawEditor で記述した翻訳規則を変換モジュールで C++の関数に変換して作成するのであるが、言語間の表現の対応によっては、現在の jawEditor では記述しきれない翻訳規則もでてくる [3]。そのような場合には、現在は変換モジュールによって生成された翻訳規則関数を手作業で修正している。翻訳規則についての検討をさらに進め、jawEditor の記述機能を高めること、翻訳規則関数の種類を拡充することは今後の課題である。

翻訳規則関数呼び出しルーチン GenET() は、翻訳規則の木構造 TransferTree(TT) のルートノードから TT の各ノードの翻訳規則関数を再帰的に実行して、目的言語の表現構造 ExpressionTree(ET) を作り出す。

TT のノード N に対して、GenET(N) は次のような手順で N に対応するオブジェクト ET(N) を生成する。

(1) N に対して BaseType の子ノード (子$_B$) がある場合には GenET(子$_B$) を呼び出し、子$_B$ に対する表現構造 ET(子$_B$) を再帰的に生成する。

(2) N の翻訳規則関数を実行して N に対応する Base タイプのオブジェクト ET(N_B) を生成する。この際、翻訳規則関数の引数には (1) で生成した ET(子$_B$) が子ノードとして渡される。

(3) 未処理の子ノード (Addition タイプの子ノード・子$_A$) がある場合には GenET(子$_A$, ET(N_B)) を呼び出し、ET(N) を完成させる。引数の ET(N_B) は GetET 内で作成される ET(子$_A$) の親として扱われる。Addition タイプの子ノードが存在しない場合は ET(N) = ET(N_B) として GenET() を終了する。

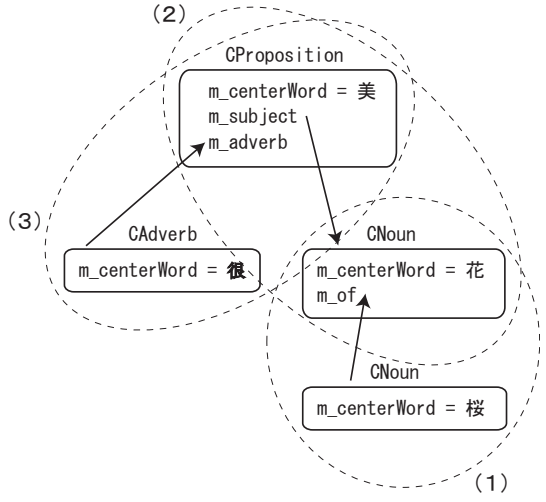

図 5: 「桜の花はとても美しい」に対する中国語の ET

3.4 機能語表現に対する翻訳規則

日本語では用言に後接する機能語で時制や否定、話者の判断など、命題的内容に対するモダリティを表現する。また体言をはじめとする様々な品詞の語に後接

し、取立て表現を構成する機能語も多い。このように膠着語である日本語では機能語は重要な役割を担っている。jaw では命題的内容を翻訳するための表現パターン翻訳規則とは別に機能語表現に対する翻訳を行うための翻訳規則を機能語翻訳規則テーブルとして RDB 上に用意している。機能語表現の翻訳は命題的内容の翻訳によって目的言語の表現構造が作成された後、機能語翻訳規則テーブルを参照し、表現構造のオブジェクトに必要な表現要素を書き込むことで行う。どのような種類の表現要素が必要となるかは目的言語によって異なるため、機能語翻訳規則テーブルも目的言語毎に設ける必要がある。

機能語翻訳規則テーブルは 2 種類ある。1 つは原言語・目的言語間で対応が 1 対 1 になる場合に用いるテーブル (タイプ A) である。タイプ A のテーブルには訳語を指定するための翻訳規則を記述する。もう 1 つは対応が 1 対多になる場合に用いるテーブル (タイプ B) である。タイプ B のテーブルには訳し分け条件と適合時の処理内容を指定するための翻訳規則を記述する。訳し分けの条件は表現構造中の要素の値に対して条件を設定するものである。

タイプA

機能語	専用テーブル名	代入変数名 1	訳語 1	···
か	—	m_modeC->m_tone	吗	
・				
・				
・				
ない	SP_nai	—	—	
・				

タイプB

ID	条件変数名	関係	条件値	代入変数名 1	訳語 1	···
01	m_centerW	==	有	m_modeC->m_negative	没	
02	m_element4	==	ば	m_modeC->m_negative	不	
・						
・						
99	—	default	—	m_modeC->m_negative	不	

図 6: 機能語テーブルの例

4　表現構造からの線状化

線状化関数は各クラスのメソッドとして定義されている。線状化関数はクラスのメンバとして表現されている目的言語の言語表現の要素や標識をもとに語順規則に沿って訳文を生成する (図 7)。

現在のところ線状化は表現構造のルートとなるオブジェクトから単純に再帰的に生成関数を呼び出すことで行っているが、生成した部分句の文字列長などを判断して表現方法を変更するといったより高度な線状化規則を実装してゆくことは今後の課題である。

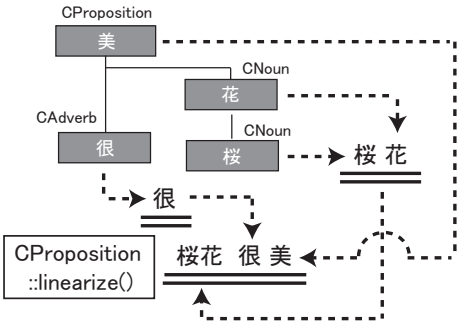

図 7: 線状化のイメージ

5　いくつかの言語への適用実験

現在我々は中国語、ベトナム語、ミャンマー語、シンハラ語、および日本の手話を目的言語とした機械翻訳システムの開発を行っている。これらの言語は中国語を除けば、日本語との間の機械翻訳の対象としてはこれまでほとんど研究されてきたことはなかった。現在それぞれ 200 文程度の例文を題材として翻訳実験を行って分析している段階である。日本語とこれらの言語との間の翻訳規則の数はまだ少なくプロトタイプの段階ではあるが、jaw を用いてこれらの言語を目的言語とするシステムを短期間に構築することが出来た。この実験については [3] で述べている。

6　おわりに

日本語から多言語への機械翻訳のためのエンジンとして作成したシステム jaw の処理方式について述べた。大域パターンへの対応や翻訳規則関数の拡充など、実際の言語間対応の分析に基づいて、さらに研究・整備を続けていく予定である。

参考文献

[1] 今井啓允 他：オブジェクト指向言語のパラダイムを利用した機械翻訳エンジン jaw
言語処理学会 第 10 回年次大会（2004)

[2] 今井啓允 他：日本語からアジア諸言語への機械翻訳の試み
情報処理学会 第 65 回全国大会（2003)

[3] 浅井良信 他：機械翻訳システム jaw と多言語への翻訳実験
言語処理学会 第 11 回年次大会（2005)

[4] 田中伸明 他：日中機械翻訳システム jaw/Chinese における取立て表現の翻訳処理
言語処理学会 第 11 回年次大会（2005)

[5] 谷口真代 他：日本語-手話機械翻訳システム (jaw/SL) 構築の試みと翻訳実験
言語処理学会 第 11 回年次大会（2005)