Internal Use Only (非公開)

TR-SLT-0020

Sandglass: 換言と言語変換の協調による機械翻訳モデル

Sandglass: Machine Translation by Interaction
between Paraphraser and Transfer

山本 和英
Kazuhide Yamamoto

2002 年 9 月 30 日

Abstract

換言処理を基軸とした機械翻訳／音声翻訳モデル (SANDGLASS 翻訳モデル) の提案を行なっている。これまでに、日中翻訳を対象に翻訳モデルの一部を実装し、プロトタイプを作成した。本稿では、このプロトタイプのうち原言語換言部と言語変換部における協調処理について報告する。

(株) 国際電気通信基礎技術研究所
音声言語コミュニケーション研究所
〒 619-0288 「けいはんな学研都市」光台二丁目 2 番地 2 TEL:0774-95-1301

Advanced Telecommunications Research Institute International
Spoken Language Translation Research Laboratories
2-2-2 Hikaridai "Keihanna Science City" 619-0288, Japan
Telephone:+81-774-95-1301
Fax:+81-774-95-1308

©2002 ATR 音声言語コミュニケーション研究所
©2002 by ATR Spoken Language Translation Research Laboratories

目次

第 1 章

はじめに

英語が翻訳対象言語でない場合、例えば中国語と日本語の間での音声翻訳について述べる。これらの言語間においては日英や英日とは異なり対訳コーパス (bilingual corpus) の入手は容易ではないため、対訳コーパスを利用した翻訳手法は有力な手法とは言えない。特に音声翻訳の場合はこの傾向が顕著であり、将来に渡って複数言語で対応づけされた大量の音声言語資源の入手は現実的ではないため、コーパス利用の手法は実用的ではない。一方、言語によっては二言語話者を確保することさえ容易ではないため、手作業による翻訳知識の構築も比較的困難である。日英／英日翻訳とは異なるこのような状況下で、我々はどのような音声翻訳モデルを考えるのが現実的かについて議論する。

幸い、日本語もしくは中国語に限定すれば、近年単言語コーパスの入手は容易となり、単言語話者に作業依頼することも比較的容易である。もし、これを有効活用して原言語および目的言語で単言語処理を十分に行なうことができれば、二言語に直接関わる知識及び処理を低減した翻訳機構を構築できるのではないかと考えた。すなわち、変換処理に基軸を置いた機械翻訳モデルを転換し、原言語と目的言語の両者の換言処理を翻訳機構の基軸に置いた機械翻訳モデルを提案する。このモデルでは、従来は変換部で解くべき問題のできるだけ多くを原言語と目的言語の換言処理の問題と捉え直し、変換部では文字通りの「言語変換」すなわち原言語表現から (複数の) 目的言語表現への写像 (候補列挙) のみを行なう。

本モデルはちょうど、我々が翻訳対象言語に不慣れな場合に行なう訳出作業に似ている。例えば、中国語の知識がほとんどない状況で中日翻訳を試みる場合、一旦は辞書で (不自然でも構わずに) 日本語に置き換えてから、日本語単独で考え、より自然な日本語に言い換える。また日中翻訳の場合は、当初は日本語入力表現に対して辞書を引くが、日中辞書に項目記載のない場合もあり、その場合は別の日本語表現に一旦言い換えて辞書を引くという作業を、我々はごく当然のように行なっている。以上の観察から、二言語知識を豊富に持つのが理想的ではあるが、仮に二言語に関係する翻訳知識がある程度不足していても、原言語並びに目的言語で換言処理を十分に行なうことができれば、ある程度の機械翻訳は可能であると考えた。

以上の基本方針に基づいて、現在我々は中日翻訳を対象にして音声翻訳システムの構築を進めている [Yam02]。本報告では、本提案機構の基本設計を述べ、翻訳方式の議論を行なう。次に、出力性能実験の結果を報告する。最後に、試作したプログラムの概略を述べ、最後にまとめを行なう。

第 2 章

SANDGLASS 翻訳モデル

2.1 各処理部の独立性と処理方略

SANDGLASS モデルにおいては、原言語における換言処理 (以下、単に換言処理と呼ぶ) と目的言語への言語変換処理 (以下、単に変換処理と呼ぶ) の独立性を高くして部品化することで各部の開発効率を高める。また同時に、換言部においてできるだけ汎用的な換言処理を行なわせることで機械翻訳に依存しない換言機構の構築を目指す。

変換部がどのような知識を持つかに依存しない換言処理を実現することは部品の汎用性や開発の独立性など様々な特長を持つが、その一方でどのような換言を行なえばいいのか、あるいはどのような表現が変換可能なのかという換言目標が立てづらい。これに対する一方策として、換言部において入力文のすべての可能な換言文を作成し、この全文に対して変換処理を試み、変換に成功した文のうち最良のものを変換結果とする方策がある。しかし、例えば実時間性を要求される音声翻訳においてこのような処理を実時間の範囲で実現することは考えにくく、また無目標で換言文を大量に生産することは合理的でない[1]。

これに対し、SANDGLASS モデルでは以下のような方策を立てた。すなわち、換言部と変換部は互いに独立した部品として構成するが、両者の間に制御部を設け、この制御部を介して換言部と変換部が協調することで翻訳文を生成する。つまり、翻訳を行なうための情報の流れは換言部から変換部への一方的なものとはせず、換言部と変換部の両者が与えられた入力文に関して自らの必要十分な情報を交換することで翻訳文を生成する枠組みを提案する。具体的には以下のような特徴を持つ。

1. 換言部は 1 回の換言要求につき必ず換言文 1 文を出力する。
2. 変換部は変換に失敗した場合、換言部に換言を行なうための「ヒント」を、可能な場合に出力する。
3. 換言部はヒントにできるだけ沿った換言文の生成を試みる。ただし、ヒントに従うかどうかは換言部が判断する。

[1] 一般的には局所的で独立性の高い換言を複数箇所で行なうことが可能な場合が多く、その結果組み合わせ的に非常に多数の換言文を生成可能である。

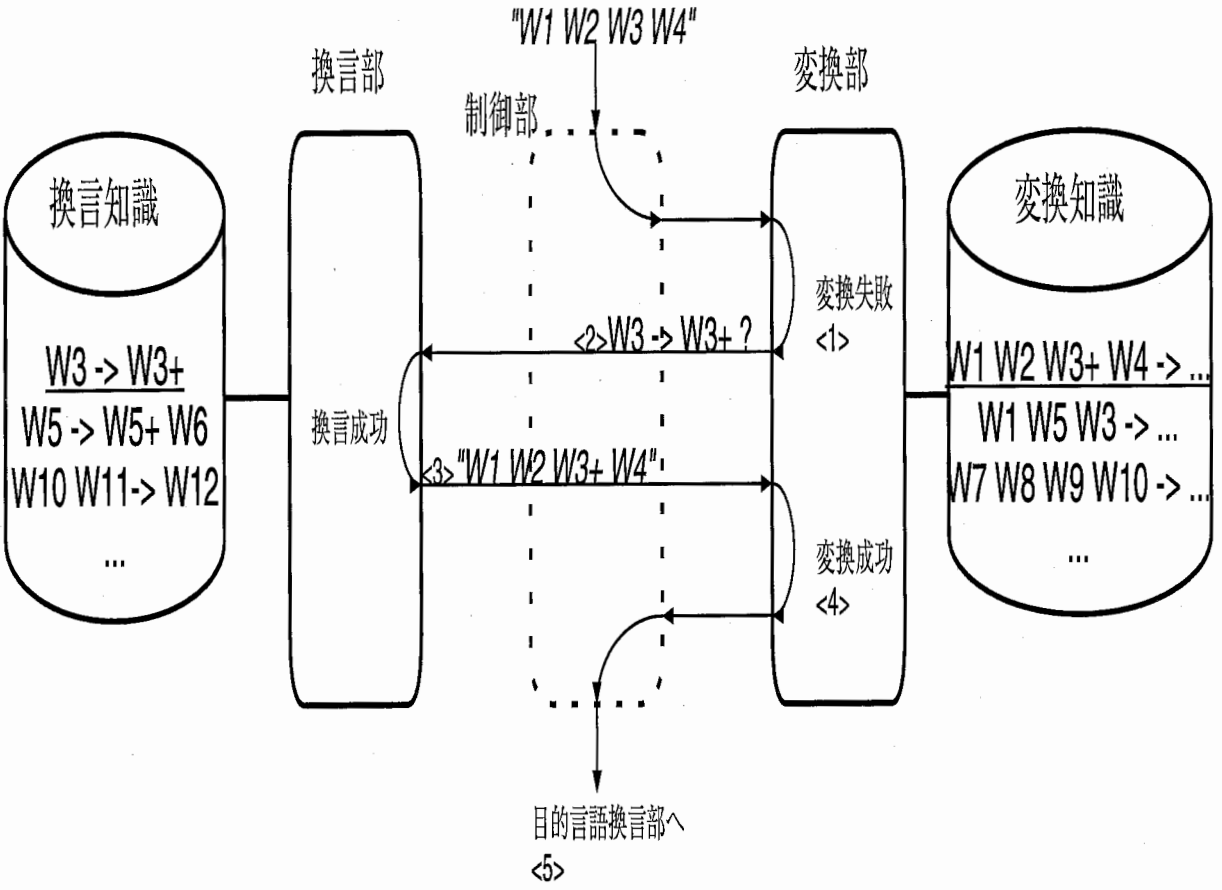

図 2.1: 換言部と変換部の協調による翻訳モデル

2.2 換言部と変換部の協調機構

図2.1に我々の提案を図示した。翻訳機構の主要部は(原言語)換言部と変換部から構成され、両者を制御する制御部がその間に入る[2]。この機構の特徴は(1)換言部と変換部が対等で処理の上流、下流という関係がない(2)換言するための換言知識と変換するための変換知識が完全に独立であり、双方が独自の基準で処理を行なう、の2点である。

形態素解析結果はまず制御部を介して目的言語への変換が試みられる。仮に、$W_1W_2W_3W_4$ という形態素列の変換を試みるが失敗したとする⟨1⟩。この際、変換可能な $W_1W_2W_{3+}W_4$ という類似した表現があり、もし W_3 が W_{3+} に換言可能であれば変換可能であるという情報を変換部が換言部に伝える⟨2⟩。この情報を受け取った換言部は換言知識を調べ、W_3 が W_{3+} に換言可能であるためこの語を換言して変換部に返す⟨3⟩。変換部は換言後の表現の変換を試み、今度は成功するため⟨4⟩、変換されて目的言語となった表現は以後の処理に渡される⟨5⟩。

仮に、最初の変換処理⟨1⟩で類似した表現を見つけ出すことができなかった場合は、換言部が自らの換言知識を使用することによって何らかの換言文を生成する。変換部から与えられた換言の「ヒント」の通りに換言できないと換言部が判断した場合も、換言知識を使って換言を試みる。

2.3 換言部

換言部では以下の種類の換言を以下の順に行なう。いずれかで換言事例を得ることができた時点で処理は終了し、換言結果を出力する。

1. 変換部からの「ヒント」による換言
2. 入力分割
3. 敬語の削除 [Oht01]
4. 文末表現などの単純化 [Yam01]
5. 名詞連続の複合名詞化
6. 文の要素削除

2.3.1 変換部からの「ヒント」による換言

変換部からの「ヒント」による換言は、変換処理の結果出力された換言候補の通りに換言することが可能かどうかの検証を行なう。この処理は、あらかじめ語彙レベルで換言可能な事例を用意しておき、変換部から与えられる換言候補との照合を行なう。この処理によって行なわれる換言は、ある1単語を同一品詞内で異なる1単語に換言するもののみである。換言される語は主に機能語であるが、内容語である場合もある。

原文 おもしろそうですねえ

換言文 おもしろそうですね

原文 御安心ください

換言文 ご安心ください

[2]本論に関係のない処理部(例えば目的言語換言部)は簡単のため省略した。

2.3.2 入力分割

入力分割は、入力が2文以上から構成される場合にこれを分割することが処理の目的である。話し言葉は書き言葉とは異なり、句点で入力を分割することが不可能であるので、終助詞などの存在で入力を分割する規則を用意した。この規則に照合した場合、分割を行なう。なお、以下では分割点を記号 ";" で表現する。

原文 それじゃあさよなら

換言文 それじゃあ; さよなら

原文 いくらですかそれ

換言文 いくらですか; それ

2.3.3 敬語の削除

待遇表現は日本語会話表現に頻出するため、このすべての表現を翻訳対象とすることは二言語知識の増大を招き、好ましくない。このため、可能な範囲で簡単化もしくは削除を行なう([Oht01])。

原文 ではいかがいたしましょうか

換言文 ではどうしましょう

原文 あいにくですがございません

換言文 あいにくありません

2.3.4 文末表現などの単純化

文末表現等も日本語には多様な表現が存在するため、これを簡単化する。また、口語的表現(「…してる」「…したんですけど」など)の削除もできる限り試みる([Yam01])。

原文 風邪じゃないかと思うんですけど

換言文 風邪でしょう

2.3.5 名詞連続の複合名詞化

名詞連続の複合名詞化は、普通名詞またはサ変名詞が直接または助詞「の」を介して連続していた場合にこれを一語の名詞(以下の例では{…}と表す)としてみなす処理である。後述する変換部で行なう処理の都合上、複合名詞と単一名詞を同一視して変換することが不可能なため、テンプレート方式の欠点を補うためにこのような処理を行なう。ただし、話し言葉の場合にこの処理を無条件で行なうことは問題があるため、処理の優先度を低くした。

原文 火曜日の午後五時なんですが

換言文 { 火曜日の午後五時 } なんですが

2.3.6 文の要素削除

文要素の削除は、最後の手段として文を構成する要素のうち、比較的重要でないと考えられる副詞(「ちょっと」「たぶん」「本当に」など)や助詞「は」「も」など(他の格助詞と同時に使われる場合)の削除を行なう。

原文 明日までにはご用意いたしますよ

換言文 明日までに用意します

原文 たぶん十分くらいだと思います

換言文 十分くらいだと思います

2.4 変換知識の作成

事前準備として、簡易の変換知識の作成を行なった。変換知識として、本研究では二言語対訳コーパスと複数の対訳が付与されている日中対訳辞書から自動作成した。

まず、我々の収集した日本語旅行会話表現約23万文に対し全文に中国語訳を付与した。ここで、中国語対訳は形態素情報を含む一切の言語情報は付与されていない。次に、JUMAN[3]とKNP[4]を使って日本語解析を行なった後で、日中対訳辞書を用いて単語単位の単純な日中の対応付けを行なった。すなわち、日本語解析結果の各単語に対して、その対応する中国語単語が中国語訳文の中に含まれているかどうかを調べ、照合した場合は日中の当該単語に同一のID番号を付与する。ある単語に対して、対訳辞書中の複数の単語に対して照合した場合は、それらのうち最長の中国語単語に対して対応付けを行なった。

ここで、ある日本語に対してある単一の中国語訳が複数箇所に見られる場合は、そのすべてに対して対応付けした。例えば、「行きませんか」に対して「去不去?」(「去」は「行く」の意)という対訳が付与されていた場合、「〈去 #538〉不〈去 #538〉?」などとして日本語の「行く」に対する同一のID(例では#538)を付与した。

以下では、この処理によって対応づけされた原言語(日本語)と目的言語(中国語)一対の表現対をテンプレート、テンプレート中で対訳辞書によって対応付けできた部分をそのテンプレートの変数、それ以外の項目を固定表現と呼ぶ。

2.5 変換部

変換部は以上の処理で作成されたテンプレートに基づく翻訳を行なう。変換処理は、テンプレート検索、テンプレート照合の二つの処理から構成される。

まず、入力文(の形態素解析結果)に対し、入力文と全く同一の品詞列を持つテンプレートを検索する。これが存在しない場合は変換に失敗し、新たな換言文を要求する。

同一品詞列のテンプレートが検索できた場合は、次に各形態素ごとに入力とテンプレートとの照合を行なう。照合にすべて成功した場合は変換成功となり、目的言語へ変換した結果を制御部に返す。テンプレート照合に失敗した場合は、テンプレート中の一部の固定表現が入力とは異なることを意味する。つまり、両者の差異を変換部が把握しており、またこの差異が換言処理の際の目標表現になり得るので、この相違する表現列を換言のための「ヒント」として制御部に返

[3] http://www-nagao.kuee.kyoto-u.ac.jp /nl-resource/juman.html
[4] http://www-nagao.kuee.kyoto-u.ac.jp /nl-resource/knp.html

す。ここで、テンプレート照合に失敗したテンプレートが複数ある場合は、それらすべてに対して差異を抽出し、列挙して返す。

第 3 章

処理例：「やはりそうだったんですか」

本提案手法において、具体的にどのような情報がどのようにやり取りされているのかについて、例文「やはりそうだったんですか」を例に説明する。

やはりd␣そうkだった/だbんです/んだxかe

日本語解析結果はまず、制御部を経由して変換部に渡し、言語変換を試みる。変換部は、入力文と同一の品詞列 (dkbxe) を持つ文を検索した結果、以下の 1 文が該当した。

なぜ#216279dそう#216280kな/だbのです/のだxかe
〈为什么 #279〉是〈那样 #280〉呢?

この 2 文を比較すると相違は「やはり –〉なぜ」「だった –〉な」「んです –〉のです」の 3 形態素である。このうち、最初の相違「やはり –〉なぜ」はテンプレートの「なぜ」が変数化されており、他の語に置換可能であるが、他の 2 語についてはテンプレート中は固定表現であることから照合せず、変換に失敗する。

この結果、変換部は制御部に「変換失敗」という情報を返す。同時に、もし「だった –〉な」「んです –〉のです」という換言が可能であれば変換可能である、という情報を以下のような形式で制御部に返す。制御部は、この換言の「ヒント」を換言部に与え、新たな換言文を要求する。換言部は、このヒントの通りに換言することが可能かどうかの検証を行なう。自ら持つ換言知識と照合した結果、「だった –〉な」の換言が可能と判断し、換言を行なう。その結果次の文を制御部に返す。

やはりd␣そうkな/だbんです/んだxかe

制御部はこの文を変換部に与え、二度目の変換処理を要求する。変換部では前回と同様の処理を行なった結果、変換は失敗だがもし「んです –〉のです」と換言できれば変換可能だという情報を返す。換言部はこの情報を利用してさらに換言を試み、変換部のヒントに従った換言に再び成功する。その結果換言部は次の文を制御部を介して変換部に渡す。

やはりd␣そうkな/だbのです/のだxかe

変換部はこの文に対して三度目の変換処理を試み、最終的に成功する。

第 4 章

実験

以上のシステムに対して、変換知識量と出力性能の関係を見る実験を行なった。変換知識は、旅行会話に関する約 23 万文の日中対話文と複数語訳が付与された日本語約 51000 項目の日中辞書を使用して対応付けを行なった。この対訳文集合から無作為に 100 文〜 23 万文の範囲で変化させた実験を 3 回行なった。評価文は、変換知識とは独立に 10 形態素以下の 1000 文を無作為に選んだ。

図 4.1に実験結果を示す。ここで、縦軸は翻訳文の出力率であり、訳質の評価は行なっていない。グラフにおいて、部分出力 ('partial') と完全出力 ('whole') の差異は主に文分割処理による貢献を、完全出力と換言処理を一切行なわない場合 ('no paraphrasing') の性能差は換言処理そのものの貢献、すなわち本稿で提案した協調処理による貢献と考えることができる。この図より、変換知識の非常に少ない場合でも換言処理の貢献によって約 20% の文を部分的に出力可能にしており、また変換知識の増大に伴って (実験の最大変換知識量程度では) 換言処理の貢献はむしろ増大している。

ところで、本研究で行なったテンプレート変換のような、形態素単位の客観的な一般化のみで得られる変換知識では、実験の最大規模 (23 万文) で 3 割程度しか未知文を翻訳できないことがわかる。すなわち、全文を出力させるためには残り 7 割の文に対する知識を何らかの手段で補わねばならないとも解釈できる。これに対して人手で作成した現在の換言知識と協調処理はこのうち約 4 割を補うことが可能であることを実験では示した。一方、このような機構なしで、すなわちコーパスから得られる情報のみを用いた汎化処理あるいは統計処理だけでこの 7 割を補うことは容易ではないと考える。すなわち、翻訳処理の際には、例えば本稿で示した換言知識のような原言語知識も重要なのではないか。

この実験において換言試行回数も測定した。この結果を図 4.2に示す。この図より、変換知識の増大に伴って換言試行回数は (対数的に) 単調減少していることがわかる。さらに詳細を観察すると、翻訳に成功しているものの多くは換言回数が 1 回または 2 回程度のもので、それ以上の換言を行なっているものの多くは最終的な翻訳結果の生成に失敗している。すなわち、換言回数は少数 (出力成功) と多数 (出力失敗) に大きく分かれているようであり、さらに分析が必要である。

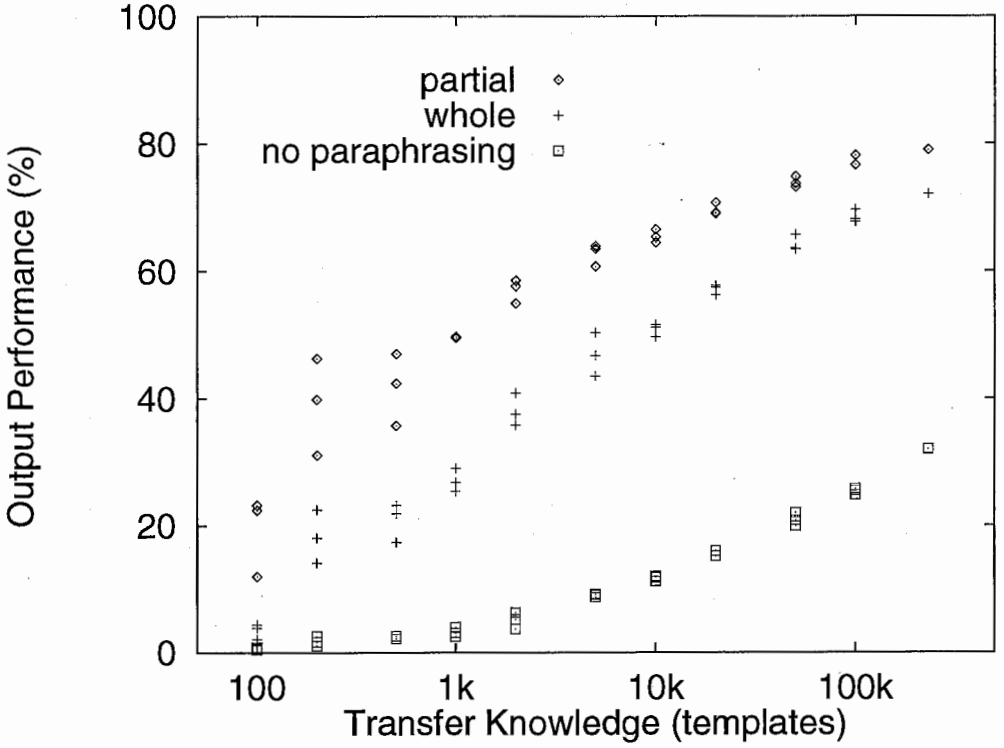

図 4.1: システムの出力性能

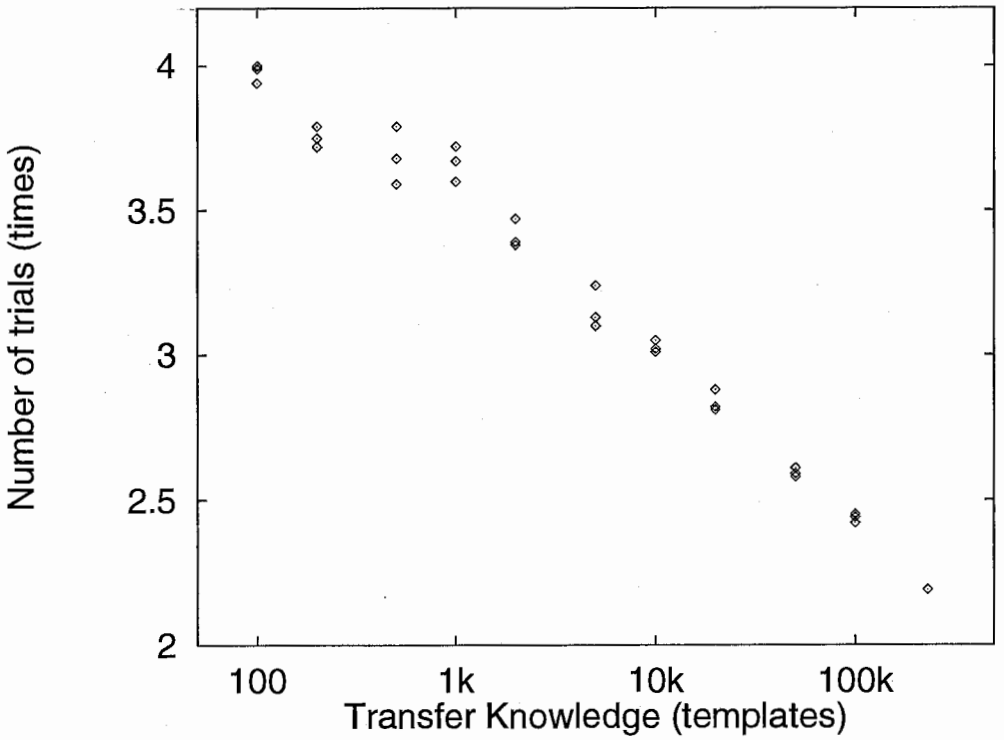

図 4.2: 換言試行回数

第 5 章

プログラム

本節では、実験的に作成したプログラムについて述べる。プログラムは perl によって作成した。
以下に、順に簡単な説明を行なう。

sandglass.pl 翻訳を実行するためのメインの起動プログラム。以下のように実行する。

```
% sandglass.pl input_file > output_file
```

現在の設定では、入力ファイルは JUMAN と KNP の実行結果を加工した形式 (以下、簡易 KNP 形式) を入力とする。1 行が 1 入力であり、日本語はいわゆる EUC 文字コード体系で記述する。以下に入力ファイルの例を示す。

```
女の子ｎです / だｂ
女将ｎさんｇです / だｂかｅ
商品ｎのｑ 受け渡しｓはｐ どうｋ 致し / 致すｖましょう / ますｔ
小さなｒ 青いａ 色ｎをｕ した / するｖ スーツケースｓです / だｂ
詳しいａ スケジュールｓはｐ もうｄ お決まりｎです / だｂかｅ
乗り換えｓし / するｖなくて / ないｔはｐ なり / なるｖませ / ますｔん / ぬｘかｅ
乗り場ｎはｐ どこｋです / だｂかｅ
場所ｎはｐ マジソンスクエアガーデンｓのｕ すぐｄ 近くｄにｕ なり / なるｖますｔ
食後ｎにｕ コーヒーｓ お願いｓし / するｖますｔ
申し訳ｓ ございませ / ございますｖん / ぬｘでした / だｂ
説明書ｎです / だｂかｅ
千二百ｎ円ｊでｕ 和食ｎとｕ 洋食ｎがｕ 有り / 有るｖますｔ
全部ｎでｕ 二十ｎドルｊにｕ なり / なるｖますｔ
全部ｎでｕ 二十ｎ名ｊです / だｂねｅ
早速ｄ おｈ持ち / 持つｖ いたし / いたすｖますｔ
滞在ｓをｕ 延長ｓし / するｖたいｔんです / んだｘがｑ
滞在ｓ先ｎはｐ ホテルニューオータニロサンゼルスｓ 六百二ｎ号ｊ室ｎ
大きめの / 大きめだａ お部屋ｎで / だｂございますｔかｅ
大人ｎ 四ｎ人ｊとｕ 子供ｎがｕ 一ｎ人ｊです / だｂ
```

プログラムの出力は 2 種類あり、一つは翻訳結果を translation というファイルに、もう一

表 5.1: 使用するファイルの一覧

ファイル名	機能
sandglass.pl	起動プログラム
demo-sandglass.pl	デモ用の起動プログラム
pm/initial.pm	初期設定ライブラリ
pm/sandglass.pm	Sandglass 翻訳処理ライブラリ
pm/transfer.pm	変換部ライブラリ
pm/vocabulary.pm	辞書検索ライブラリ
pm/correct_tag.pm	juman/knp 誤り修正ライブラリ
pm/juman_knp.pm	knp 出力形式変換ライブラリ
pm/paraphrase.pm	換言処理部
pm/honor.pm	敬語の換言部
pm/replace.pm	ヒント検証による換言部
pm/segment.pm	文分割による換言部
pm/simplify.pm	文の簡単化による換言部
pm/final_deletion.pm	文内表現削除ライブラリ
pm/generate.pm	生成部ライブラリ
pm/coding_system.pm	emacs-mule と日中のコード変換
pm/chinese_tool.pm	中国語処理ツール群
pm/japanese_tool.pm	日本語処理ツール群
pm/tool.pm	小さなツール

つは途中結果の状況 (途中の換言結果など) を標準出力に、それぞれ出力する。

demo-sandglass.pl デモ用の起動プログラム。任意のタイプ入力に対して日中翻訳を試みる。
起動の方法は単に、

```
% demo-sandglass.pl
```

とタイプするだけである。これを実行すると、必要なファイルの読み込みが終わった後に自由入力可能な状態になる。実行は日中の文字を混在表示するために emacs/Mule 上の shell 環境で行なう。この際、emacs/Mule において set-buffer-process-coding-system を実行し、プロセスの入出力を共に文字コード emacs-mule (*internal*) に設定する。実行の様子を以下に示す。

```
pxs014[21:54]~/project)./demo-sandglass.pl

------------------------------------------------------------------------

          Sandglass: Japanese-Chinese Machine Translation

                              Kazuhide Yamamoto
        (c) 2001 ATR Spoken Language Translation Laboratories

------------------------------------------------------------------------

  Reading Lexicon...
  Reading Japanese Corpus...
  Reading Chinese Corpus...

Sandglass% こんにちは
  Chinese% 您好。

Sandglass% 明日のシングルを予約したいのですが
  Chinese% 我想预订明天的单人。

Sandglass%
pxs014[21:55]~/project)
```

ここで、「こんにちは」「明日のシングルを予約したいのですが」がタイプ入力した部分である。このプログラムは、入力文を受け付けた後に juman と knp をプログラム内部で逐次呼び出しているため、実行の際には juman と knp が必須である。

pm/initial.pm 初期設定ライブラリ。sandglass.pl から呼び出される。対訳辞書の読み込み、日本語コーパスの読み込み、中国語コーパスの読み込みの三つの作業を行ない、読み込んだ変数を返す。

pm/sandglass.pm 翻訳処理ライブラリ。上記 sandglass.pl から呼び出される。論文における制御部に相当する。これまでにどのような換言が行なわれたかを保存し、変換結果に応じて、換言部、変換部を順に呼び出す。また、文分割の処理に伴い、換言部を呼び出す際に、分割の各単位ごとに呼び出して統合するという処理もここに含まれる。

pm/transfer.pm 変換処理を行なうライブラリ。 sandglass.pm から呼び出される。文分割されている各「文」に対し、その文の品詞列と同一のテンプレートを検索し、それらに対して入力との照合を行なう。照合の結果、単語列部分に差異がある部分はそれをヒントとして換言部に渡すという、一連の処理をすべてここで行なう。

pm/vocabulary.pm 辞書引きをする際に、対象語が数字である場合は特別な処理を行なう。そうでない場合は対訳辞書を引く。

pm/correct_tag.pm juman と knp の解析結果を修正するライブラリ。影響の大きな誤りや、juman 体系 (形態素解析辞書) と Penn Chinese 体系 (対訳辞書) との間の形態素／品詞体系のずれのうち影響の大きいと考えられるものを観察によって抽出し、規則として加えている。例えば、「はい」はほとんどの場合動詞「はう」と解析され、「ありがとう」は多くの場合、「あり (名詞) が (助詞) とう (動詞)」と解析されるため、このような誤りを修正している[1]。ただ、この規則数は最小限にしているため、現在はそれほど多くない (15 程度)。perl のパターンマッチングによって実現している。

pm/juman_knp.pm juman と knp の出力形式に対して加工や整形を行なう処理部。この中には、全品詞をアルファベット 1 文字へ写像する処理、 knp の出力形式から依存関係を抽出する処理が含まれる。品詞からアルファベットへの写像する表の一覧を表 5.2に示す。

pm/paraphrase.pm 換言処理の主処理部。ここから各換言処理部を呼び出す。ある換言処理部において換言結果が得られた場合には、その結果を変換部に渡す。いくつかの換言処理部の呼び出しを行なっているが、呼び出しの優先順位はここで規定される。

pm/honor.pm 敬語に関する換言を行なう関数類。詳しくは [Oht01] を参照。

pm/replace.pm 換言処理部のうち、「ヒント」による換言の検証を行なう。同一品詞の 1 単語による換言が可能と考えられる対を登録することによって、ヒントがここで検証される。ただし、現在ではこの知識の吟味が不十分なため、主に自動的に獲得した単語対のみを登録している。

pm/segment.pm 換言処理部のうち、文分割を行なう処理部。大きく分けて、自動的に分割箇所を決める処理と、人間が与えた規則によって分割箇所を決める処理の両者がある。ただし、前者の自動分割処理は、現状で分割精度が不十分なため呼び出しは行なわない。

pm/simplify.pm 文の簡単化に関する換言を行なう関数類。詳しくは [Yam01] を参照。

pm/final_deletion.pm 換言処理の最終段階で、文内の不要な要素の削除を行なう処理部。この中には、以下の処理が含まれる。

- 敬意の接辞の削除 (「お…」「…さん」など)

[1]juman の形態素辞書を修正することによっても実現可能かもしれないが、juman を外部ツールとして固定化するため、あえてこのような方策をとった。

表 5.2: juman の品詞と写像される記号の対応

品詞	記号
名詞	n
サ変名詞	s
動詞	v
判定詞	b
助詞	p
接続助詞	q
終助詞	e
格助詞	u
助動詞	x
形容詞	a
副詞	d
接頭辞	h
接尾辞	t
名詞性名詞接尾辞	g
名詞性名詞助数辞	j
接続詞	c
指示詞	k
連体詞	r
感動詞	i
特殊	z

- (「…までは」「…には」における) 助詞「は」の削除
- (「…では」「…での」における) 助詞「で」の削除
- その他の助詞の削除、例えば、「…にも」の「も」の削除「…だけの」の「だけ」の削除など
- 助動詞の削除、例えば「…みたいです」「…のです」など
- 終助詞の削除、例えば「…ね」「…よ」など
- その他の接辞の削除、例えば「…くらい」「約…」など
- 副詞の削除、例えば「ちょっと」「いろいろ」「たぶん」「ほんとに」など

これらはいずれも文内において削除された場合に最も影響の小さい表現群であると考え、他の手段で換言が生成できない場合にやむを得ない方策としてこれら表現の削除を行なう。

pm/generate.pm 生成処理を行なうライブラリ。sandglass.pm から呼び出される。現在は複数の訳語候補の中から最も高頻度のものを選択する処理など、いくつかのヒューリスティックスが実現されていて、その後に出力結果の表示のための整形を行なう。

pm/coding_system.pm emacs/Mule 上の emacs-mule (*internal*) コードと euc-japan コードとの相互変換、emacs-mule (*internal*) コードと euc-china コードとの相互変換を行なう。以上の 4 関数のライブラリ。

pm/chinese_tool.pm 中国語に依存する処理のツール群。出力のための整形、句読点の追加や削除、算用数字から漢数字への変換など。

pm/japanese_tool.pm 動詞を入力として、「ます形」を出力するツール。

pm/tool.pm いくつかの小さなツール類。

第 6 章

まとめ

機械翻訳問題の多くを原言語の換言問題とするための機械翻訳モデルの一事例を示した。本稿の機構に従えば、大量で高品質の変換知識を期待できない多言語翻訳などの状況であっても、限定的な変換知識を原言語の換言知識である程度補うことが可能と考えている。現在実装している換言処理はまだ十分なものとは言えないため十分な翻訳精度を得ることができないが、扱う換言現象の増大に伴って翻訳精度も向上していくと考える。

参考文献

[Oht01] OHTAKE, K. and YAMAMOTO, K.: Paraphrasing Honorifics, In *Proc. of NLPRS2001 Workshop on Automatic Paraphrasing: Theories and Applications*, pp. 13–20 (2001).

[Yam01] YAMAMOTO, K.: Paraphrasing Spoken Japanese for Untangling Bilingual Transfer, In *Proc. of NLPRS2001*, pp. 203–210 (2001).

[Yam02] YAMAMOTO, K.: Machine Translation by Interaction between Paraphraser and Transfer, In *Proc. of COLING2002*, pp. 1107–1113 (2002).