知識ベース型翻訳のための領域知識の構成法

武田 浩一
日本アイ・ビー・エム株式会社 東京基礎研究所
神奈川県大和市下鶴間 1623-14
takeda@trl.vnet.ibm.com

1 まえがき

機械翻訳 (MT) では一般的に技術マニュアルのような制限された分野で成功を収めてきたといえるが、これは注意深く作成されたユーザ辞書が文中に現れる単語のかなりの範囲を高精度にカバーするためである。改訂版などのように以前とあまり変わらない文書を翻訳する際には用例処理なども含めて、最も高い翻訳精度が期待できる。しかし、このようにユーザ辞書に強く依存したシステムでは、ユーザが新たな分野で翻訳を行なう際のカスタマイズ作業が大変であり、適合する専門用語辞書が存在しない場合には相当な量の未知語を登録する必要に迫られる。さらに、多義語の訳し分けや係り受けの優先度などを指定できるだけの記述力をもったユーザ辞書はまだ少数である。

現実に翻訳すべき文書は、同じ分野 (例えば計算機分野) に属すると考えられていても、"line" のように配線 (ハードウェア)、行 (ソフトウェア)、直線 (CAD) のように最もよく用いられる語義・訳語がすべて異なる頻出単語もあり、高精度の翻訳にはこういった問題[1]を解決することが不可欠である。

このような各分野内の下位分野ともいえる文書ごとの微妙な差異は、従来ではシステムの提供する学習機能、研究レベルでは文脈処理 [5] や用例に基づく翻訳 [6, 7] で扱われることが多かった。例えば、Translation Manager/2[3] のような翻訳支援エディタでは、階層的なユーザ辞書や翻訳用例の管理と、これらの資源の優先度順の検索といった機能が提供されているが、ある文書が与えられた時にそれがどのユーザ辞書や翻訳用例に最も関連があるかを調べるのはユーザの責任であるし、翻訳機能がないためあくまで人間による翻訳を支援する機能にとどめられている。MT システムで提供される学習機能には、訳語の優先度や係り受けの優先度を学習することができるが、これらの獲得された知識を、1つの関連した集合として再利用するという考え方はなく、単純な仮名漢字変換のように最後に指定された訳語や係り受けを第一候補として記憶しているにすぎない。用例による翻訳は、複雑な辞書項目や変換規則を用いずに、こなれた訳を生成できるという潜在的な能力があるが、複数の翻訳分野から集められた用例から満足な翻訳結果を得られるかどうか、また必要な用例のサイズ、文脈を考慮した用例の扱い[2]といった点が明らかでない。

本論文では、ユーザの指示により得られた知識を翻訳中の文書と対応づけることにより、翻訳すべき新たな文書が与えられた時に、効果的に今までに得られた知識を利用してその文書を翻訳する手法を提案し、これを RBKC (Revision-Based Knowledge Construction, *Revision* に基づく領域知識の構成法) と呼ぶことにする。RBKC では、*revision* と呼ぶ知識の集まりを用いる。*revision* には、未知語に相当する新たな辞書エントリ、特定の語義に対する優先度、係り受けの優先度、特定の単語に対する訳語の優先度、の4種類の情報が含まれ、MT システムの後編集時にユーザの指示により学習できる典型的な情報を想定している。獲得された知識は、翻訳中の文書と対応づけられているため、単に意味のある、ひとまとまりの知識として扱えるだけでなく、新たに翻訳する文書と、*revision* に対応する文書とを比較することで、柔軟に関連の強い *revision* を構成し、翻訳に利用することが可能である。

2 *Revision* に基づく知識獲得

revision とは、以下の4種類の要素の順序付きリストである。

1. 未知語に対する辞書エントリ
2. 特定の語の語義の優先度
3. 2つの句の係り受けの優先度
4. 特定の語の訳語選択の優先度

これらの情報は通常 MT システムの翻訳結果の後編集時に、ユーザが修正のために指定する情報であるか、対話的翻訳システム [4, 1] が直接ユーザから指示されるような情報に相当する。*revision* は、それが獲得され

[1] 単純な選択制約はあまり有効ではないことが多い。例えば、user line と command line は「行」の意味になり、communication line は「回線」、witness line が「線」の意味であることを選択制約で記述するぐらいなら、それらを複合語で登録するほうが容易である。

[2] 例えば、"Now, the line is available." という文を翻訳する場合の line の訳語は、この文と用例の類似度に基づく訳よりも、前の文で使われた line の訳語を用いた方がよい。

た時の翻訳文書と対にして管理される。*revision* $R = \langle e_1, \ldots, e_k \rangle$において、要素 e_i は e_jに先行するといい、$e_i \prec e_j$と書く $(1 \leq i < j \leq k)$。*revision* 内の要素の順序は、ユーザによって指定された順序を反映するものであり、各要素は、それに先行するすべての *revision* 内の要素より高い優先度をもつ。2つの *revision* は、それらが別の文書と対応づけられている場合には、一方の *revision* の要素は他方の *revision* の要素とは比較不能である。このため、*revision* 集合に対して線形順序を与え、異なる *revision* に含まれる要素の優先度を、同様に定義する。

revision の例:

```
{ ("machine translation" (lex ((cat n)))
   (sense 1)   "機械翻訳" (lex ((cat n)))),
  ("display" (lex ((cat n))) (sense 2)),
  (<"share", (lex ((cat v) (subcat trans)))>
   (prep with) <"user", (lex ((cat n)))>),
  ("function" (lex ((cat n)))
   "機能" (lex ((cat n)))) }
```

この *revision* の最初の要素は "machine translation" という英語の名詞複合語に対する辞書エントリで、語義番号が1で、日本語の "機械翻訳" という訳語がつけられている。2番目の要素は、名詞 "display" の2番目の語義が他の語義よりも高い優先度をもつことを示す。3番目の要素は、前置詞句 "with user(s)" が、他動詞 "share" に係る優先度が、他の係り先候補に係る優先度よりも高いことを示す。最後の要素は、名詞 "function" には、訳語 "機能" が最も高い優先度をもつ[3]ことを示している。

revision は単なる順序付きリストなので、同一の要素が繰り返し現れることがある。このようなケースを排除するために、以下のように正規形の *revision* を定義する。

定義 1 ある *revision* Rに対する正規形の *revision* R_cとは Rに含まれる同一の要素のうち、最後の要素を残して、他の要素をすべて除いたものである。

性質 1 Rが与える未知語の辞書エントリと、語義、係り受け、訳語選択の3種類の優先度は、R_cが与えるものと同一である。

従って、本論文では以後正規形の *revision* のみを考えることにする。

定義 2 今、$R_1 = \langle e_{11}, \ldots, e_{1m} \rangle$および$R_2 = \langle e_{21}, \ldots, e_{2n} \rangle$を2つの *revision* とすると、$R_1$と R_2の合成を以下のように定義する。

- 順序付き和:$R = R_1 \cup R_2$は$\langle e_{11}, \ldots, e_{1m}, e_{21}, \ldots, e_{2n} \rangle$からなる正規形の *revision*[4]である。
- 順序付き積:$R = R_1 \cap R_2$は以下のような要素 e からなる *revision* である。$\{e | e \in R_1$かつ $e \in R_2$, $e_i \prec e_j$ *iff* $e_i \prec e_j, (e_i, e_j \in R_1)\}$
- 順序付き差:$R = R_1 - R_2$は以下のような要素 e からなる *revision* である。$\{e | e \in R_1$かつ $e \notin R_2$, $e_i \prec e$ *iff* $e_i \prec e_j, (e_i, e_j \in R_1)\}$

これらの合成操作は可換ではないが、結合則は成り立つ。

定義 3 *revision* R_1は、以下が成り立つ時に *revision* R_2に包摂されるといい、$R_2 \sqsubseteq R_1$と書く。

1. $e \in R_2$ならば $e \in R_1$
2. R_2の任意の2つの要素 e_1, e_2に対し、$e_1 \prec e_2$ ならば、R_1においても $e_1 \prec e_2$である。

性質 2 順序付き和は単調ではない。すなわち、2つの *revision* R_1および R_2に対し、$R = R_1 \cup R_2$は必ずしも R_1に包摂されない。

性質 3 順序付き積および順序付き差は単調である。すなわち、2つの *revision* R_1および R_2に対し、以下が成立する。

1. $R_1 \cap R_2 \sqsubseteq R_1$
2. $R_1 - R_2 \sqsubseteq R_1$

revision の間の包摂関係は、あまり強力でなく、同じ単語に同一の語義や訳語を第一候補にする保証さえない。包摂関係で成り立つのは、*revision* Rが、ある単語の2つの語義や訳語に優先度を与えていれば、その関係が、Rに包摂される任意の *revision* で保存されることであり、Rに含まれない語義や訳語に高い優先度を与えることを妨げない。

3 *Revision* の柔軟な構成

翻訳された文書の量が増えるにつれて、*revision* も同様に増加する。蓄積された *revision* は新しい文書の翻訳に効果を表し始める。この際に蓄積された *revision* のうち、どれをどのような順序で利用するかを決定する必要がある。もちろん、ユーザが文書の性質をもとに、いくつかの *revision* を選ぶことは可能であるが、システムによって自動的に計算させる方法があることが望ましい。このような *revision* のリストを *revision* リストと呼ぶことにする。

[3] ある単語の語義の優先度と、その訳語の優先度を指定するような競合する2つの要素が *revision* に含まれる場合には、この順序付きリストの中で先行する要素が無効になる。

[4] Rは、要素の重複により m+n よりも少ない要素しか含まないかもしれない。

定義 4 L を k 個の *revision* $R_i, (i = 1, \ldots, k)$ から構成された *revision* リスト $\{R_1, R_2, \ldots, R_k\}$ とする。このとき L は、$R = R_1 \cup R_2 \cup \ldots \cup R_k$ に相当する *revision* を与える。

revision リストは、1 つのユーザ辞書とは異なり、動的に再編成できることが大きな特長である。例えば、特定の *revision* を無効にすることは、*revision* リストから、その *revision* を除くことに等しい。除かれた *revision* は、別の文書の翻訳の際にはまた利用可能であり、ユーザから一旦獲得した知識が失われることはない。このような柔軟性が多分野翻訳や、文脈依存翻訳にとって非常に重要である。冒頭に述べたような計算機分野の中に、ハードウェアやソフトウェアの下位分野が存在するといった階層性は、計算機分野一般の *revision*、ハードウェア分野の *revision* といった順序、あるいは計算機分野一般の *revision*、ソフトウェア分野の *revision*、という順序で *revision* を構成することで無理なく扱える。

もう 1 つの重要な特長は、以下のように既に得られた *revision* から構成される *revision* リストの質を評価する目安が存在することである。

1. *revision* リスト中の各 *revision* に含まれる要素をもとにし、要素の種類別に[5]次のような競合グラフを作成する。
   - 語 w の辞書エントリに対応する要素に対し、語を示す節点 N_w と、その辞書エントリの素性 f を表現する節点 N_f を作り、N_w から N_f への、lex というラベルをもつ有向枝で結合する。
   - 語 w の語義 s を優先する要素に対し、語を示す節点 N_w と、語義 s を示す節点 N_s を作り、N_w から N_s への、sense というラベルの有向枝で結合する。
   - 句の係り受けを優先する要素 ($\langle w_j, lex_j \rangle$ role $\langle w_i, lex_i \rangle$) に対し、$\langle w_i, lex_i \rangle$ を表す節点 N_r と、$\langle w_j, lex_i \rangle$ を表す節点 N_l を作り、N_r から N_l への role というラベルをもった有向枝で結合する。
   - 語 w の訳語 t を優先する要素に対し、語を示す節点 N_w と、訳語 t を示す節点 N_t を作り、N_w から N_t への、trans というラベルの有向枝で結合する。
2. 各競合グラフにおいて、次のような競合枝の数の総和 C を求める。
   - 語 w を示す節点 N_w から、同じラベル (lex, sense, trans) で、異なる素性、語義、訳語を示す節点を結合している有向枝の数。
   - 句 w を示す節点 N_r から、同じ role で、異なる句を示す節点を結合している有向枝の数。

[5] 2 節で述べたように、語義と訳語の優先度が競合するといった場合を考慮するためには、異種の要素からなる同様なグラフを作成する必要があるが、簡単のためここでは省略する。

直観的には、競合枝の総和 C は、多品詞語、多義語、係り受けや訳語の曖昧さを示している。C は潜在的な曖昧さではなく、ユーザが正解として指定した品詞や語義の範囲での曖昧さなので、このような C の値が大きいということは、その *revision* リストが内容的にかなり異質な *revision* を含んでいるという可能性がある。実際に、Gale 他 [2] あるいは那須川 [5] の報告ではパラグラフや章といった単位での語義や訳語の強い一貫性が観察でき、全体としてこのような分野の *revision* を集めても、C の値は小さいことが期待できる。上記の計算で、句の係り受けの曖昧性はやや過大に評価されている。例えば、

```
(<"share", (lex ((cat v) (subcat trans)))>
 (prep with) <"user", (lex ((cat n)))>)
(<"provide", (lex ((cat v) (subcat trans)))>
 (prep with) <"user", (lex ((cat n)))>)
```

のように前置詞句 "with (the) user" の 2 つの係り先 "share (something)" や "provide (something)" が競合枝とされてしまう。このようなケースは次の前置詞句が動詞句あるいは名詞句にかかるという係り先の曖昧性よりもかなり稀である。

```
(<"share", (lex ((cat v) (subcat trans)))>
 (prep with) <"process", (lex ((cat n)))>)
(<"resource" (lex ((cat n)))>
 (prep with) <"process", (lex ((cat n)))>)
```

のように "share (the) resource with (the) process" のような文の係り受けの曖昧性も競合枝として正しく反映される。従って、C の精度を高めるためには、上記の競合グラフで同一の N_r に対して、競合する N_l になる句の素性に対して制限を加えるか、重み付きの総和によって調整するのがよい。

性質 4 $L = \langle R_1, \ldots, R_k \rangle$ を *revision* リストとする。このとき競合枝数の総和 C は、L の要素の任意の並べ換えによって得られる *revision list* において不変である。

性質 5 任意の 2 つの *revision* リスト L_1 および L_2 に対し、それぞれの競合枝数の総和 C_1 および C_2 は単調である。すなわち、両者をあわせた *revision* リスト $L\ (= L_1 \cup L_2)$ の競合枝数の総和 C は、$C \geq C_1 + C_2$ を満足する。

以上の性質により、最適な *revision* リストの構成は、競合枝数の総和をできるだけ小さな revision の集まりを求め、次にそれらの線形順序を求めるという、2つのステップで決定できることがわかる。最初のステップは、以下のように記述できる。

1. 今 d を翻訳したい文書とし、$D = \{d_1, d_2, \ldots, d_k\}$ を既に翻訳された文書のうち、任意の i、j $(1 \leq i \leq j \leq k)$ に対し $WORDS(d, d_i) \geq WORDS(d, d_j)$ が成立するように並べた文書のリストとする。ただし、$WORDS(d, d_i)$ は、d および d_i で共通に現れる異なり単語の数を示す。[6]

2. h を許される競合枝数の上限とする。

3. $D_p = d_1, d_2, \ldots, d_p$ を、D の最初の p 個の文書で、それらに対応する *revision* の競合枝数の総和が h 以下であるような最大の p とする。このとき、D_p が求める *revision* の集まりを与える。

第2のステップは、次のような *revision* の対ごとの比較に基づき *revision* リストを計算する。

1. ステップ1で求めた競合グラフにおいて、同じ節点を始点とする k 個の競合枝 a_1, a_2, ..., a_k があるとする。

2. もし、この競合枝のうち、$a_i (1 \leq i \leq k)$ が今翻訳する文書 d にとり最も望ましいものであるとき、a_i を含む *revision* は、それ以外の a_1, ..., a_k を含む *revision* に先行されなければならない。

3. すべての競合枝の始点となる節点に対し、上記の *revision* の対に関する順序を決定し、これらの順序と矛盾しない *revision* 全体に対する1つの線形順序があれば、これを解として *revision* リストを構成する。このような線形順序の計算は、対ごとに指定された順序の総和に対して多項式時間で計算できる。また、もとの順序が巡回的にならない時、およびその時に限り、望む線形順序が存在する。

もしこのステップで *revision* の線形順序が存在しない場合には、望ましい要素のみを残した1つの *revision* を新たにこの文書に対応づけるができる。ただし、この場合には、*revision* と、もとの文書の対応関係が失われてしまうので、柔軟に *revision* を再構成することはできなくなる。

[6] このような類似度の計算は、Yarowsky[8] でも有効であることが示されている。また、一方にしか含まれない単語の数をペナルティとして評価するように修正することも可能である。

参考文献

[1] C. Boitet and H. Blanchon. "Dialogue-Based MT for monolingual authors and the LIDIA project". In *Proc. of the Natural Language Processing Pacific Rim Symposium '93*, pages 208–222, Fukuoka, Japan, Dec. 1993.

[2] W. Gale, K. Church, and D. Yarowsky. "One Sense Per Discourse". In *Proc. of the 4th DARPA Speech and Natural Language Workshop*, 1992.

[3] IBM Corp. "IBM SAA AD/Cycle Translation Manager/2 Release 1.0". Technical report, SH12-5906, 1992.

[4] H. Maruyama, H. Watanabe, and S. Ogino. "An Interactive Japanese Parser for Machine Translation". In *Proc. of the 13th International Conference on Computational Linguistics*, pages 257–262, Helsinki, Aug. 1990.

[5] T. Nasukawa. "Discource Constraint in Computer Manuals". In *Proc. of the 5th International Conference on Theoretical and Methodological Issues in Machine Translation*, pages 183–194, Kyoto, Japan, July 1993.

[6] S. Sato and M. Nagao. "Toward Memory-based Translation". In *Proc. of the 13th International Conference on Computational Linguistics*, pages 247–252, Helsinki, Aug. 1990.

[7] E. Sumita and H. Iida. "Experiments and Prospects of Example-Based Machine Translation". In *Proc. of the 29th Annual Meeting of the Association for Computational Linguistics*, pages 185–192, Berkeley, June 1991.

[8] D. Yarowsky. "Word-Sense Disambiguation Using Statistical Models of Roget's Categories Trained on Large Corpora". In *Proc. of the 14th International Conference on Computational Linguistics*, pages 454–460, July 1992.