O-011

翻訳メモリ共有による翻訳支援 Web アプリケーションの開発

Translation Support Web Application by Sharing Translation Memory

岡田正彦† 奥野 拓‡

Masahiko OKADA Taku OKUNO

1. まえがき

翻訳作業を支援，効率化するために「翻訳メモリツール」と呼ばれるソフトウェアが存在する．このソフトウェアは，「翻訳メモリ」と呼ばれる原文と訳文が対になったデータベースを利用し，そこに登録されている翻訳と同じ，または類似の原文が翻訳対象の原文に存在する場合，翻訳メモリから自動的に訳文を引用することにより，翻訳作業の効率化と質の向上を支援するものである．現状ではその特徴から主にプロの翻訳家やソフトウェアのローカリゼーションなどで利用されている．これらは非常に有用なソフトウェアであるが，初期導入時コストの高さなどから一般のユーザにはあまり認知されておらず，積極的に利用されているとはいえない．

本研究の目的は，翻訳メモリツールを利用して一般のユーザでも気軽に効率よく翻訳できる環境の提供とそこで生成される翻訳メモリや用語集などの翻訳資産を有効活用できるアプリケーションの実装である。以下では，既存の翻訳メモリツールの特徴と問題点，そしてその解決手法とアプリケーションの実装について説明する．

2. 翻訳メモリツール

2.1 翻訳メモリツールの効果と種類

翻訳メモリツールは，過去に訳した文章を引用，参照しながら翻訳作業を進めていくため，以下のような効果が期待できる．

- 過去に翻訳した文章を何度も翻訳するといった無駄を省く
- 過去に翻訳した類似文を検索する手間を省く
- 類似文や同一文に対する訳文の表現に統一感を持たせる
- 翻訳品質を向上させる

また，翻訳メモリには TMX [1]や XLIFF [2]といった XML 形式の標準ファイルフォーマットが存在しているため，翻訳者自身による翻訳データが蓄積された翻訳メモリだけでなく，他の翻訳者が翻訳した翻訳メモリとの併用や異なる翻訳メモリツール間での翻訳メモリの再利用が可能である．

さらに，ほとんどの翻訳メモリツールでは特定分野の専門用語や任意の単語の対訳をまとめた用語集の登録が可能であり，用語集に登録済みの用語が翻訳対象文に出現した場合に，自動的に提示，または置換するような機能を持っている．

翻訳メモリツールは，スタンドアロン型と Web 型に大別することができる．前者はクライアント PC 上の翻訳メモリを利用して個人で翻訳作業を行うためのものであり，後者は Web サーバ上の翻訳メモリを利用して複数人で共同翻訳を行うためのものである．現状最も広く利用されているものはスタンドアロン型の Trados [3]に代表される商用の翻訳メモリツールであるが，近年はスタンドアロン型の OmegaT [4]などオープンソースの翻訳メモリツールが開発されており，その利用も広がってきている．

スタンドアロン型，Web 型それぞれについての長所と短所を表 1 に示す．

表 1 種類別翻訳メモリツールの長所と短所

種類	長所	短所
スタンドアロン型	翻訳作業に特化したユーザインタフェースと豊富なショートカット	共同翻訳の管理には別ツールを利用
	ローカルの辞書データなどとの連携	翻訳資産の共有は手作業
Web 型	共同翻訳の管理が容易	翻訳作業時のユーザインタフェース，操作性がスタンドアロン型に劣る
	翻訳資産の共有が容易	

2.2 翻訳メモリツールの問題点

翻訳メモリツールが一般ユーザに積極的に利用されていない原因として主に以下のような問題点が存在する．

(1) 翻訳メモリ・用語集の事前準備

翻訳メモリツールは，機械翻訳のように原文から訳文を自動生成するのではなく，あくまでも既存の翻訳資産から検索された類似訳例を参考にすることで翻訳作業の効率化を図るツールである．そのため，翻訳メモリツールの利用によって作業効率化を実感するためには，ある程度翻訳対象と関連のある翻訳メモリや用語集を事前に準備しておく必要がある．

(2) 翻訳メモリの入手

現状，翻訳メモリは翻訳会社やオープンソースソフトウェアの翻訳プロジェクトなどに分散して存在しており，利用者はその中から必要分を探し出し，取捨選択しなければならない．また，必要とする翻訳メモリが存在しない場合には，既存の対訳文から翻訳メモリの形式に自力で変換しなければならない．

(3) 翻訳メモリに訳例が存在しない場合

もしも事前に準備した翻訳メモリ内に翻訳対象文と類似した訳例が存在しなかった場合，多くの翻訳メモリツールでは何も参照することができない．よって，結局一から翻訳することとなり，翻訳メモリツールを利用している意味がなくなってしまう．

これら以外にも，前述の Trados などが非常に高機能な故に高価であることも利用されていない原因の一つであると考えられる．

3. 提案手法

3.1 翻訳資産のタグによる分類と活用

翻訳メモリツールを利用して効率的な翻訳作業を行うためには，ある程度の分量と適切な内容の翻訳メモリや用語集が必要である．そのため，初期導入時において期待する効果を得るためには，前述のとおりそれらを翻訳者自身が事前に準備しておかなければならない．また，その翻訳作業の結果得られる翻訳メモリについても，現状では分散して存在しているため，有効に活用できていないという問題がある．これらの問題点については，翻訳メモリなどの翻訳資産を集中管理することによって解決することが可能であると考えられる．しかし，翻訳資産の増加によって文書のカテゴリが多様化することで，一つの原文

†公立はこだて未来大学 システム情報科学研究科

‡公立はこだて未来大学 システム情報科学部

に対して多種多様な訳例が提示される可能性があり，その選択次第では原文本来の意味と全く異なる訳文になってしまうことも考えられる．極端な例として，小説と情報系の技術文書といった訳文の体裁も表現方法も全く異なるような分野の訳例が混在している場合がある．

この問題については，ユーザが翻訳対象文書の持つ特徴や分類に基づいて付与したタグと，用語集を参照し，その一致率や一致数によって，より適切な順に訳例を並べ替えて提示することでその解決を図る．この時付与されるタグは，文書のカテゴリを表す「必須タグ」とユーザが文書のキーワードなどから独自に考案し，複数付与することのできる「自由タグ」の2種類で構成される．自由タグのようにユーザ独自のタグ付与を許可した場合，全く同じ文書に対してでもユーザによって全く異なるタグを付与する可能性があるため，本来，高い関連性を持つはずの文書同士においても完全に関連性のないものとして扱われる可能性がでてきてしまう．必須タグのような最低限のカテゴリ分けをすることで，このような事態の発生を減少させることができると考えられる．今回必須タグについては，Wikipediaの主要カテゴリ及びその1階層下のサブカテゴリを用いる．

用語集については翻訳対象文書中に含まれる専門用語やキーワードを登録している可能性が高く，関連性の高い文書同士であれば共通する可能性が高いことから並べ替えの基準として利用することが可能と考えられる．例えば，「bug」という単語にはコンピュータの「バグ」や「昆虫」など分野によって複数の訳し方が存在する．この「bug」を「バグ」として用語集に登録している文書が複数存在していれば，それらは情報系の文書であり，ある程度の関連性を保持している可能性がある．極端な例でいえば，前述した小説や情報系の技術文書といった，一見全く異なるように見える文書同士でも「bug」の訳し方が同じであれば，ある程度の関連性を保持していると考えられる．

3.2 翻訳作業におけるヒントの追加

訳例が存在しない場合には何も参照できる情報がないという問題については，翻訳作業時に参照できるヒントを追加し，それらをシームレスに参照可能にすることにより問題解決と作業の効率化を図る．従来のヒントは，事前に準備した翻訳メモリ内から検索された訳例と用語集のみであったが，今回はそれに加えて以下の3項目を追加する．

（1）辞書の参照

翻訳作業中に最も参照する頻度の高い情報が単語や用語の意味であり，これらをシームレスに参照可能にすることで手間を省き，より効率的な翻訳作業が可能になると考えられる．

（2）機械翻訳結果

機械翻訳結果は，あまり精度は高くないが，翻訳作業において有用な情報であり，参照可能にすることで作業効率が向上すると考えられる．さらに，翻訳メモリに訳例が存在しない場合についても，最低限，何も情報を参照できない状況は回避できる．

（3）提示される訳例の前後文脈

意訳や特殊な翻訳例などが存在する場合，訳例の前後文脈を参照可能にすることで，その訳例が創出された状況を確認することができ，意訳に対する理解を補助することができると考えられる．

4. アプリケーションの実装

3.1及び3.2で提案した手法をWebアプリケーションとして実装した．翻訳者は翻訳対象文書にカテゴリとタグを付けて登録し，対訳エディタ上で翻訳作業を行う（図1参照）．対訳エディタ上での訳文入力時には，通常の訳例と同時にWeb上の機械翻訳サービスから取得した翻訳結果が表示される．また，翻訳対象原文中の単語を選択するとWeb上の辞書サイトから単語の意味を検索した結果がポップアップ表示される．

今回，クライアント側の対訳エディタをFlex [5]で，サーバ側のシステムをDjango [6]とMySQL, OmegaTの訳例検索エンジンの組み合わせで実装した．このようにWebアプリケーションとして実装することにより，Web型翻訳メモリツールの長所である翻訳メモリの容易な共有と共同翻訳における容易な管理が可能になった．またエディタ部分をFlexで実装することにより，Web型の欠点であったユーザインタフェースの改善が可能になり，Webブラウザ上でもスタンドアロン型と同じようにリッチな操作感を実装することが可能となった．そして翻訳作業の結果生成される翻訳資源を一カ所に蓄積し，それを利用することにより，翻訳メモリツールの初期導入時における翻訳資産準備の手間を省き，訳例不足の解消を図った．

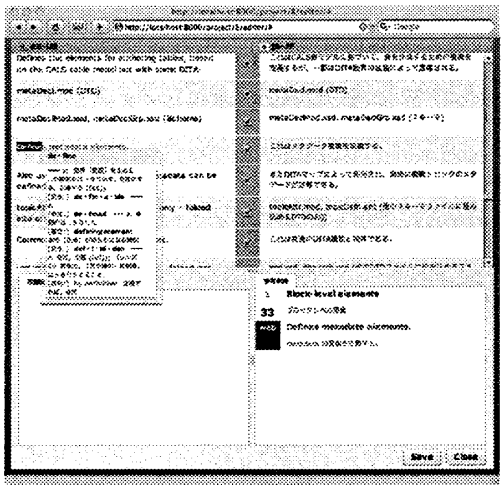

図1　提案アプリケーション（翻訳中の画面）

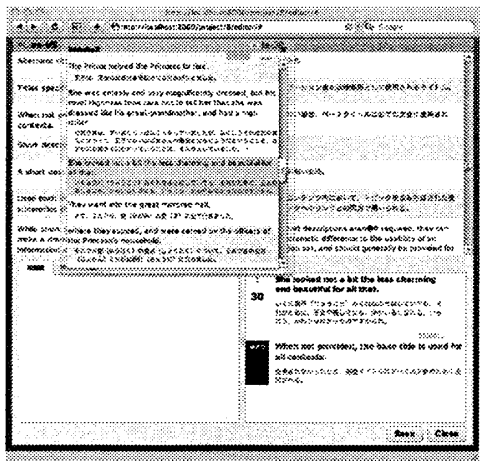

図2　提案アプリケーション（前後文脈参照画面）

5. 評価

翻訳作業におけるヒントの追加による翻訳作業効率化の検証を目的とし，翻訳にかかる操作時間を従来の翻訳メモリツール

と比較した．

5.1 評価方法

提案アプリケーションとOmegaTの両方で，同一文書について同分量の翻訳作業を行い，それぞれ翻訳完了までにかかった操作を記録，KLM手法 [7]を用いて表2のように操作を時間に換算した上で比較した．

表 2　KLM手法のオペレータと実行時間

オペレータ	内容	時間(秒)
K	キーボード・マウスボタンを押す	
	1キー・マウスボタンを押す	0.2
	複数のキーを同時に押す	0.28
P	マウスポインタを動かす	
	$T=50+150\log_2(D/S+1)$ D:カーソルから目標までの距離[px] S:目標の大きさ[px]	T
H	ホームポジション⇔マウス間移動	0.4

5.2 評価結果

図3に両ツールの翻訳作業完了までの操作合計時間とその内の辞書参照操作時間を，図4に各オペレータにおける両ツールの操作時間を示す．図3より，翻訳作業完了までの操作合計時間について提案アプリケーションの方がOmegaTよりも約20%減少していることがわかる．特にその中の辞書参照操作の時間については約30%の減少が見られるが，図4よりマウスが関連した操作時間について約10%増加していることがわかる．

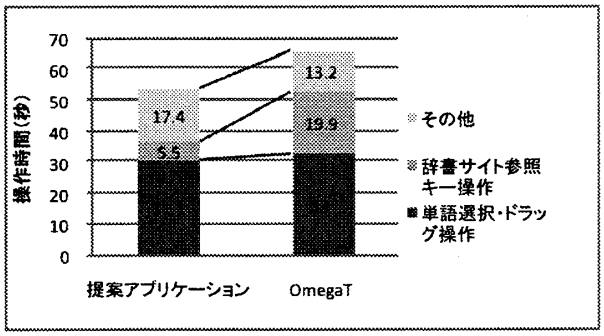

図 3　操作合計時間と辞書参照操作時間

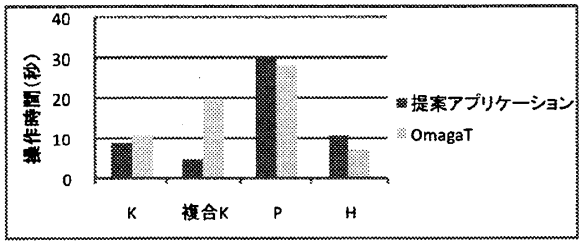

図 4　各オペレータの操作時間

6. 考察

評価結果より，辞書参照操作についてはある程度翻訳作業の効率化が実現できたと考えられる．しかし，マウスによる操作時間及びマウスとキーボード間の移動時間については逆に増加してしまっている．マウスとキーボード間の移動が増加することは連続した動作を妨げ，大きなストレスを与える可能性がある．この問題に対しては，キーボードショートカットの実装などにより改善が可能であると考えられる．

7. 今後の課題

7.1 操作性の向上

評価結果から得られた情報を基にしたキーボードショートカットの実装と，より快適な翻訳作業のためのユーザインタフェースの改良が必要である．

7.2 誤訳への対策

多くの翻訳メモリや用語集が登録されることにより，必然的に増加することが予想される誤訳への対策が必要である．具体的には，構文解析の利用による文法間違いの排除，フレーズ検索結果の利用[7]による最も適切な表現の発見と各表現への重み付けや句レベルでの対訳対の発見とその利用[8]による訳例一致箇所の増加と自動挿入による入力の効率化などを検討している．

7.3 その他の課題

前後文脈参照についての有用性の検証や翻訳対象文書からのキーワード自動抽出によるタグ関連操作の自動化などが考えられる．

8. 結言

本研究では，翻訳資産のタグによる分類と翻訳作業におけるヒントの追加により，翻訳メモリツールの問題点解決とその有効利用を図った．

最終的には，本アプリケーションをオープンソースソフトウェアとして公開し，多くのユーザに利用してもらうことにより，翻訳資産の有効活用を可能とし，現状の翻訳環境を改善することが期待される．

参考文献

[1] LISA: Translation Memory eXchange (TMX), http://www.lisa.org/Translation-Memory-e.34.0.html.

[2] XLIFF 1.1 Specification, http://www.oasis-open.org/committees/xliff/documents/cs-xliff-core-1.1-2031031.htm.

[3] SDL Trados, http://www.sdl.com/en/products/products-index/sdl-trados/.

[4] OmegaT, http://www.omegat.org/omegat/omegat_en/omegat.html.

[5] Flex, http://www.adobe.com/jp/products/flex/.

[6] Django | The Web framework for perfectionists with deadlines, http://www.djangoproject.com/.

[7] Card S.K., Moran T.P., Newell A.: The keystroke-level model for user performance time with interactive systems. Communications of the ACM, 23(7), pp. 396-410, (1980).

[8] 大鹿広憲，山名早人，検索エンジンを使った英作文支援システムの構築，2004年度修士論文，早稲田大学大学院，(2004).

[9] 荒牧英治，黒橋禎夫，佐藤理史，渡辺日出雄，用例ベース翻訳のためのパラレルコーパスからの対訳対発見，情処研報，NL-144-4, pp. 23-30, (2001).