RJ-003

表形式の言語資源からのメタデータの半自動生成

Semi-automatic Generation of Metadata from Language Resources in Table Formats

石松 昌展† 田仲 正弘‡ 石田 亨†
Masanobu Ishimatsu Masahiro Tanaka Toru Ishida

1. はじめに

現在 Web 上には多言語辞書や用例対訳集などの表形式の言語資源が，多数蓄積されている．これら多数の表から自動的に Web サービスを作りだすことができれば，機械翻訳などの他の言語サービスと組み合わせることができる．例えば，日英医療用語集から Web サービスを作成し，それを日英機械翻訳と連携させると，英語圏の外国人旅行者が日本の病院に訪れた場合，受付での支援をすることができる．このような活動が言語グリッドプロジェクト内で行われており[1]，本研究もこのような活動から動機づけられたものである．言語グリッドプロジェクトでは，世界の 90 以上の団体が異文化コラボレーションのために言語資源を共有し，活動を行っている．

Web 上にある多言語で記述されたコンテンツには，辞書や Wikipedia など様々なものがある．これらのコンテンツも膨大な量の対訳を有しているが，表はその中でも構造が詳細に記述されており，その構造により表中の情報の関係が明確に表されている．本研究では抽出した対訳を用いて Web サービスを作成することを目的としているため，対訳の対応関係が明確である表を用いた．本研究を応用して，Wikipedia などの多言語コンテンツから対訳を抽出することもできると考えられる．

表形式の言語資源から Web サービスを作成し，それを他の Web サービスと連携できるようにするためには，言語資源からメタデータを生成する必要がある．このような，表からメタデータを生成する手法についてはこれまでに様々な研究で提案されてきた．代表的な手法としては，先見的に与えられた表のモデルを用いるもの[2,3]や，知識ベースを用いて内容を判断するもの[4,5]，多数の学習例からの一般化によるモデルを用いるもの等がある[6,7]．

しかしながら，これらの手法は表形式の言語資源からメタデータを生成するのには適していない．なぜならば，同一の表形式の中に多様な意味構造を表現することができるため，あらかじめ定めたモデルを適用したり，機械学習のために多数の学習例を用意したりすることが困難であるからである．

このような背景を踏まえたうえで，本研究では，人間の例示に基づいて表構造を解釈し，メタデータを生成する手法を適用してこの問題を解決する[8]．この手法では人間が表の一部の解釈を例示として与える．表構造において，同一の構造の中には同種のメタデータが存在していると想定すると，例示の繰返しを表全体から探し出すことでその表を解釈することができる．このように，表構造を与えられた例示の繰り返しで表現することを表構造の一般化という．

しかしながら，この手法を表形式の言語資源の解釈に適用するには，二つの課題がある．一つめの課題は，人間の例示に基づく表構造の一般化の誤りである．例示を基に繰り返し構造を探索する際に，例示と等しい構造を持つにも関わらず，異なる意味構造を有する部分があった場合，システムにはその違いを見抜く能力がない．このため，システムは誤ったメタデータを生成してしまう．二つ目の課題は，ユーザによる例示の誤りである．先行研究では，例示の繰返しを探索することで表構造を解釈することから，ユーザは正確な例示を与えなければならない．もし，ユーザによる例示の中に誤りがあると，一般化は失敗してしまう．しかしながら，例示を与えることはすべてのユーザにとって容易であるわけではない．なぜならば，先行研究では，例示を与えるために RDF や表構造の一般化アルゴリズムに関する専門的な知識が必要とされるためである．

上記のような二つの課題を解決するために，我々は表形式の言語資源に潜在する制約を利用する．制約には例えば，言語資源は通常 1 つの多言語表現の組（以下，対訳と呼ぶ）の繰返しで表現されるというものがある．この制約を用いることで，我々は例示の中の繰り返されるべき部分を判別することができる．さらに，我々は制約を用いてメタデータの種類を制限することができた．このように制約を用いた処理を実現することで，我々は既存の一般化手法を拡張することができた．また，本研究で提案した手法を Web 上にある 50 の言語資源を用いた実験によって評価した．

2. 言語資源の観察

2.1 対象とする言語資源

本研究が対象とする言語資源は Web 上に蓄積された表形式の言語資源である．これらの言語資源は，以下の二つの点を満足するものとする．一つ目は Web 上で構造記述言語を用いて表構造が表現されていることである．二つ目は，ある表現や単語が複数の言語で表現されていることである．例えば，図 1 の表では英語と日本語で川の名前が表現されている．我々はこれ以降この表を用いて詳細を説明する．

日本語	英語	備考
	あ行 戻る	
アスワン	Aswan	滝:Nile川の滝
アマゾン河	Amazon	河:Andes山脈～ブラジル
ウラル川	Ural	川:ウラル山脈～カスピ海
オリノコ川	Orinoco	川:ベネズエラ

図 1 表形式の言語資源[1]

† 京都大学大学院情報学研究科，Graduate School of Informatics, Kyoto University
‡ 独立行政法人 情報通信研究機構 言語グリッドプロジェクト，Language Grid Project, National Institute of Information and Communications Technology
[1]http://www.rondely.com/zakkaya/dic3/kawa.htm

我々は Web 上に蓄積された表形式の言語資源に含まれている情報を観察した．すると，表形式の言語資源は 4 つの構成要素からなることがわかった．4 つの構成要素とは，翻訳元・翻訳先言語（*source/target languages*），対訳文（*original/translated expressions*），属性名（*attribute names*），属性値（*attribute values*）である．表 1 において，翻訳元・翻訳先言語は一行目に示されている．日本語が翻訳元言語であり，英語が翻訳先言語である．対訳文には原文（*original expression*）と翻訳された表現（*translated expression*）がある．表 1 では 1 列目と 2 列目に示された，日本語や英語による川の名前がこれにあたる．同じ意味を表す語が多言語で表現されている．この多言語による対訳文の組を対訳（*multilingual tuples*）と呼ぶ．対訳は言語資源の主要な構成要素である．属性名は，対訳に付随する情報の種類を表す．例えば表 1 の 1 行目の「備考」は属性名である．属性値は属性の値である．通常属性名と同じ行や列のセルに記述される．表 1 では 3 列目の最初の行に属性名が記述され，同じ列の 3 行目から 5 行目にかけて属性値が記述されている．また，2 行目の「あ行」も索引という属性の属性値であるが，このように属性名が表中に記述されていない属性値も存在する．

2.2 表形式の言語資源の構造

表形式の言語資源の構造を，後述のセルの間の構造概念に基づいて形式化する．ここで用いる構造概念は，セルの隣接，同じ行や列のセル，同じセルの構造の繰返しである．まず，セルの隣接に着目する．二つの隣接するセルが異なる幅や高さを持っている場合，それらには異なる種類の情報が記述されていると考えることができる．すなわち，隣接するセルの幅や高さを比較することによって，セルの構造を解析することができる．例えば表 1 の 2 行目のセルは周りの隣接するセルの 3 倍の幅を持っている．このセルには索引の属性値が記述されており，周りの対訳に関する情報が記述されたセルとは異なっている．次に，同じ行や列のセルに着目する．同じ行や列のセルに記述されている情報は，同種の情報を持つ傾向がある．表 1 のそれぞれの列を見ると，列ごとに同じ種類の情報が記述されていることが分かる．また，表の中に同一の構造を持つセル（または複数のセルからなるユニット）が複数現れた場合，それぞれのセル（ユニット）には同じ種類の情報が記述されていると考えられる．例えば，表 1 の 3〜6 行目は同じ構造をもっており，同じ種類の情報を持っていると考えられる．実際，それらにはすべて対訳が記述されている．これらの繰り返し現れる同じ構造を，繰り返し構造と呼ぶ．

2.3 言語資源の持つ制約

言語資源を観察し，我々は言語資源に潜在する制約を見出した．この制約を，RDF（Resource Description Framework）と OWL（Web Ontology Language）を用いて記述した．図 2 は表 1 の言語資源の持つ制約を RDF を用いて記述したもので，図 3 は OWL を用いて記述したものである．表からメタデータを生成する既存の手法では例示として図 2 のような RDF による記述を与えていることから，本研究では言語資源の制約を説明するために RDF を用いた．また，OWL は，RDF に現れるプロパティの制約を記述することができる言語であるため，言語資源の持つ制約を記述するために OWL を用いた．表 1 には例示として与えられる RDF による記述で用いられるプロパティの説明が，表 2 には，OWL で言語資源の持つ制約を記述する際に用いたプロパティ制約の説明がそれぞれ列挙されている．

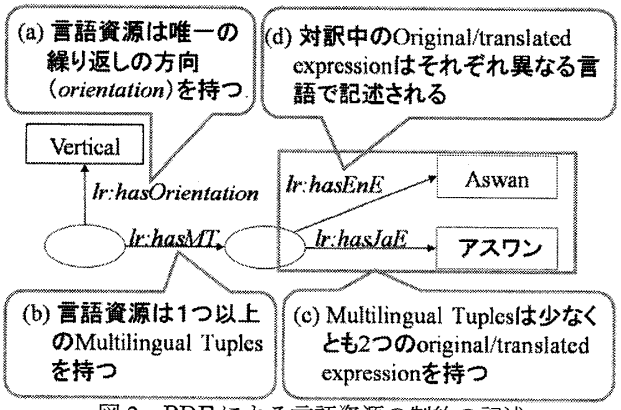

図 2 RDF による言語資源の制約の記述

表 1 プロパティ

Property	Explanation
hasOrientation	言語資源は唯一の繰り返しの方向（Orientation）を持つ
hasMT	言語資源は対訳（multilingual tuples）を持つ
hasME	対訳は多言語の対訳文（multilingual expressions）を持つ
hasJaE	対訳は日本語の対訳文を持つ．
hasEnE	対訳は英語の対訳文を持つ

表 2 プロパティ制約

Constraint	Explanation
cardinality	プロパティの数はＮである（Ｎは定数）
minCardinality	プロパティの数はＮ以上である（Ｎは定数）
allValuesFrom	プロパティの全ての値は指定されたクラスに属している

言語資源に潜在する制約には大きく分けて二つの種類がある．それは，言語資源一般が持つ制約（General constraints）で，もうひとつは各言語資源に特化した制約（Domain-Specific constraints）である．すべての言語資源は言語資源一般が持つ制約を満たしているが，各言語資源に特化した制約はそれぞれの言語資源に対して与えられる．以降，言語資源に潜在する 6 つの制約について，これらの図表を用いて説明する．

まず 1 つ目の制約は，表形式の言語資源が唯一の繰り返しの方向（orientation）を持つことである．表形式の言語資源が繰り返し構造を持つことは 2.2 節で述べたが，その繰り返しには方向があることがわかった．例えば，図 1 の表では列ごとに繰り返されているため，繰り返しの方向は図 2 に示すように縦（Vertical）となる．この制約はそれぞれの図の（a）に対応している．2 つ目は、言語資源が 1 つ以上の対訳（multilingual tuples）を持つことである．これは図中の（b）に対応している．表形式の言語資源の主構成要素は対訳であるため，言語資源は 1 つ以上の対訳を含んでいなければならない．3 つ目は 1 つの対訳が少なくと

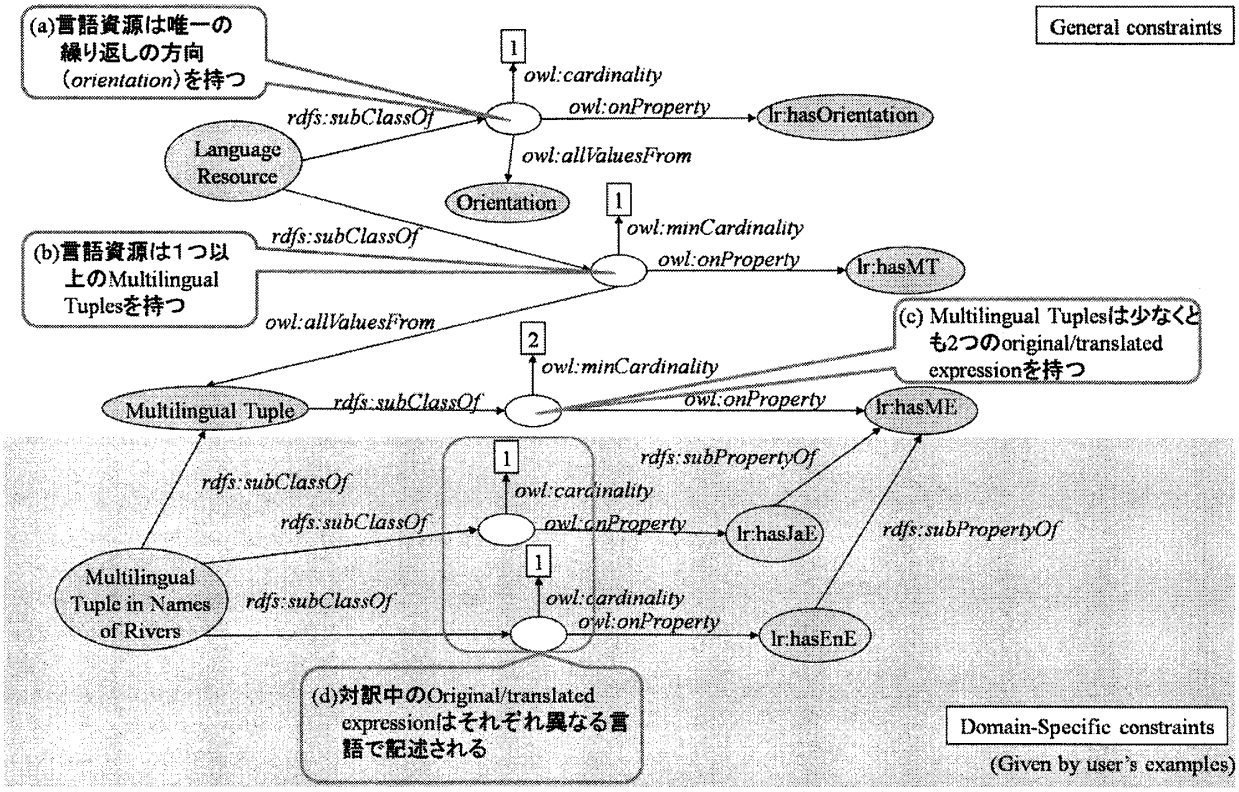

図 3　OWL による言語資源の制約の記述

も 2 つの対訳文を持つことである．対訳は多言語による対訳文の組であるので，その数は 2 以上であることは明らかである．これは図中の（c）に対応している．4 つ目の制約は，対訳中の対訳文（original/translated expressions）はそれぞれ異なる言語で記述されていることである．これは図中の（d）に対応している．5 つ目は，言語資源が 1 種類の対訳によって構成されることである．例えば，図 1 の言語資源は，日英の対訳という 1 種類の対訳のみを含んでいる．最後の制約は言語資源が対訳の繰返しで表現されることである．これらの制約から，表形式の言語資源が 1 種類の対訳の繰り返しによって構成されることが分かる．

3. 言語資源の持つ制約に基づくメタデータ生成

3.1 例示に基づくメタデータの半自動生成

この章では表形式の言語資源からメタデータを半自動生成する手法について提案する．まず，既存の手法[8]を言語資源に応用した本手法の処理の流れを図 4 に示す．この処理は，ユーザが HTML で記述された表形式の言語資源を入力することで開始される．表形式の言語資源は，処理の中で容易に扱えるようにするため，セルの縦、幅、座標、抽出した情報のみから構成されるモデルへと変換され，簡単化される．それぞれのセルの座標は，行と列の数字と対応している．図 1 の言語資源を変換して得られたモデルを図 5 に示す．

次にユーザは言語資源の表の一部分に対し，その部分の情報に意味的な解釈を与えるために例示を行うように求められる．図 6 の RDF の記述は，図 1 の言語資源に対するユーザの例示を示している．これらの例示は，言語資源の 1 行目から 3 行目に対する意味的解釈を示している．プロセスは，この例示を一般化することで，表構造を解析することになる．

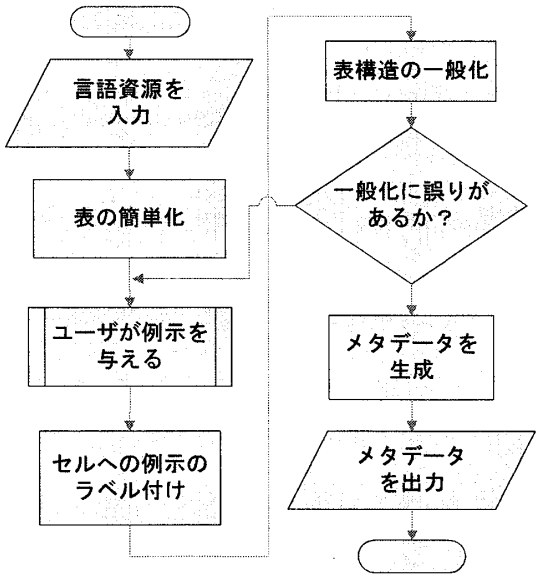

図 4　処理の流れ

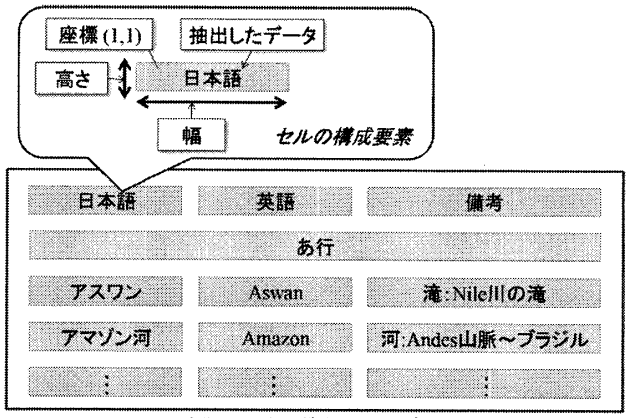

図 5　言語資源のモデル

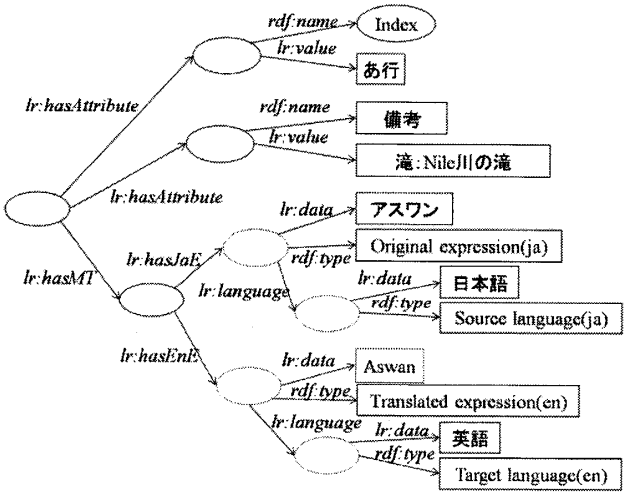

図 6　RDF による例示

2.2 節で述べたように，同じ行や列にあるセルは，同種の情報を持つ傾向がある．そこで，表を列や行を構成するセルの集まり（ユニット）に分割することを考える．ユニットが行を構成するものか列を構成するものになるかは，表の繰り返し構造の方向によって決まる．たとえば，図 1 の場合，繰り返しの方向は縦とすると，ユニットは行を構成するように分割されることになる．ユニットは行（または列）を構成するが，ユニットの中には複数の行（または列）にまたがるものもある．それは，ユニットの中に複数の行（または列）にまたがるものが存在する場合である．我々は，これらのユニットに含まれるセル同士は密接な関係を持っていると考える．たとえば，図 1 の表のユニットの場合，それぞれのユニットは行を構成するように分割されるが，それらは対訳のまとまりを示していたり，属性値のみのまとまりであったりする．このため，これらのユニットは，意味的なつながりの観点から，これ以上小さなユニットに区切らないとする．このように，行や列を構成するように表を分割して得られたユニットを，原子ユニットと呼ぶ．言語資源において，原子ユニットは対訳を表していることが多い．図 1 の表においても，3 行目から 6 行目を構成するそれぞれの原子ユニットは対訳ごとのまとまりになっている．また，1 行目や 2 行目のように，属性名や言語名のみ，属性値のみといったものを表す原子ユニットも存在する．複数のユニットの集まりを複合ユニットと呼ぶ．複合ユニットは，原子ユニットと複合ユニットから構成される．このように，ユニットは再帰的に表現される．図 7 は図 1 の言語資源をユニットに分割したものである．

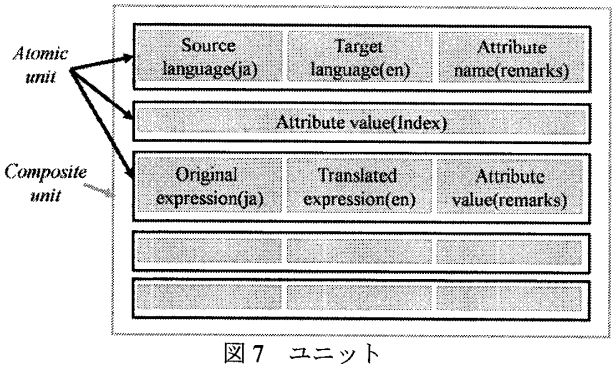

図 7　ユニット

表構造を解釈するために，プロセスは表のモデルとは別に，表をユニットで表現したものを生成する．これは，表の構造から繰り返しを発見する際に簡単に処理が行えるようにするためで，それぞれのユニットに含まれるセルは，縦，横，座標の情報しか持たない．プロセスはまず，表を原子ユニットに分割する．これらすべての原子ユニットから構成される複合ユニットは，表全体の構造を表している．次に，ユーザによって与えられた例示に基づき，例示が与えられたセルの意味的解釈をラベル付けする．この時点で，原子ユニットに含まれるセルには，初期の情報のほかに，意味的解釈がラベル付けされたものが存在することになる．図 7 において，1 行目から 3 行目を構成する原子ユニットに含まれるセルには，図 6 の例示によって与えられた意味的解釈がラベル付けされている．

次に，意味的解釈を与えられたユニットは自動的に一般化される．一般化プロセスの入力は，先述の表全体を表す複合ユニットである．プロセスはその複合ユニットに含まれるユニットから，意味的解釈がラベル付けされたセルを含むものを 1 つずつ抜き出し，そのユニットと同じ構造を持つユニットを探す．もし抜き出したユニットと同じ構造を持つ連続したユニットが見つかると，プロセスはその連続したユニットを，抜き出したユニットの繰り返しを表す新たなユニットと置き換える．置き換えが行われると，表全体の構造を表す複合ユニットは更新されたことになる．そこで，この複合ユニットは，一般化プロセスの新たな入力となる．こうして，この複合ユニットは，置き換えがなくなるまで繰り返し一般化プロセスの入力となり，置き換えが行われなくなったとき，我々は一般化された表構造を得ることができる．一般化された表構造は，原子ユニットや複合ユニットの繰り返しで表現される．例えば，図 7 を一般化すると，図 8 のようになる．複合ユニットに含まれる“[ ]+”はその間にあるユニットの繰返しを表している．

最後に，言語資源からメタデータを生成する．プロセスは，表のモデルの構造と一般化構造を比べる．たとえば，図 5 の 1 行目と図 8 の 1 行目を比べる．すると，「日本語」は「Source language(ja)」，「英語」は「Target language(en)」，「備考」は「Attribute name(remarks)」というメタデータが得られる．同様に，図 5 の 4 行目から，「アマゾン」は「Original expression(ja)」であるといったメ

タデータを得ることができる．図 9 に，図 5 の表のモデルと図 8 の一般化構造から得られるメタデータを示す．

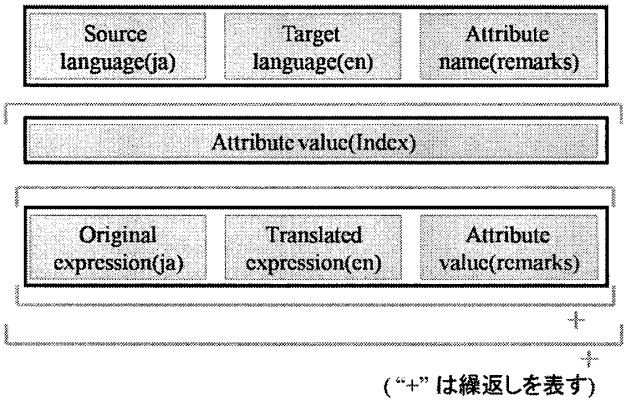

図 8　図 7 の一般化構造

```
Multilingual Tuples1:
  Original expression(ja): アスワン川
  Translated expression(en): Aswan
  Attribute value(remarks): 滝：Nile 川の滝
Multilingual Tuples2:
  Original expression(ja): アマゾン河
  Translated expression(en): Amazon
  Attribute value(remarks): 河：Andes 山脈～ブラジル
Multilingual Tuples3:
  Original expression(ja): ウラル川
  Translated expression(en): Ural
  Attribute value(remarks): 川：ウラル山脈～カスピ海
Multilingual Tuples4:
  Original expression(ja): オリノコ川
  Translated expression(en): Orinoco
  Attribute value(remarks): 川：ベネズエラ
```

図 9　生成されたメタデータ

3.2　言語資源の持つ制約の利用

既存の手法を用いて言語資源からメタデータを半自動生成するためには 2 つの課題がある．1 つ目の課題は，表構造の解釈の一般化において誤りが生じる場合があることである．表の中には，異なる意味的解釈を持つが同じ構造を持つようなユニットが存在することがある．既存の手法では構造のみの比較で同じものであるかどうかを判断するため，このようなユニットは同じだとみなされてしまう．この課題を解決するために，我々は表形式の言語資源の制約を用いた一般化アルゴリズムを提案する．このアルゴリズムは図 4 の構造の一般化の部分で用いられる．2 章で述べた表形式の言語資源の持つ制約の中に，言語資源は 1 種類の対訳の繰り返しによって構成され，一つの対訳の中の対訳文はそれぞれ違う言語で記述されるという制約がある．図 10 にこれらの制約を用いた繰り返し構造発見のためのアルゴリズムを示す．このアルゴリズムの入力は，図 7 で示した表全体を表す複合ユニット，*generalized_table* と，一つの対訳を構成するメタデータのラベルの集合である *required_labels* である．一般化の際，*required_labels* は繰り返されるべきユニットを選ぶのに使用される．これは，言語資源が 1 種類の対訳の繰り返しによって構成されるという制約によるものである．繰り返されるユニットはテーブルに含まれる対訳を表すメタデータを全てラベルとして含んでいなければならない．例えば，図 1 の言語資源の *required_labels* は，対訳が日本語と英語の対訳文から構成されるため，「Original expression(ja)」と「Translated expression(en)」である．

アルゴリズムでは，まず *generalized_table* の中の部分配列を列挙することによって，部分配列の集合を得る．これらの部分配列は，原子ユニットと複合ユニットから構成される．次にプロセスは，その部分配列の中から繰り返されるべきユニットを探す．このユニットは先に述べた二つの制約を満たす．そこで，アルゴリズムでは，短い方から順に部分配列が制約を満たすかどうかをチェックする．

```
Inputs: generalized_table /* a whole table with metadata labels
                as shown in Figure 7*/
        required_labels /*a set of metadata labels for representing
                a multilingual tuple */
Output: generalized_repetitive_structure /* a generalized_table
                representing a generalized repetitive structure as
                shown in Figure 8*/
discoverRepetition(generalized_table, required_labels){
  for each subsequence s of generalized_table (from short to
                long subsequences){
    if all labels in required_labels are contained in s
      if s iteratively appears in generalized_table
        generalized_table ← replace iteration of s in
        table with s+
        /*s+ is a composite unit which represents one or
        more iteration of s */
      return with discoverRepetition
                (generalized_table, required_labels)
    else s← the next shortest subsequence}
  return with generalized_table }
```

図 10　繰り返し発見のアルゴリズム

アルゴリズムの中の最初の if 文は，言語資源が対訳の繰り返しで構成されるという制約に基づいたものである．この制約に基づけば，繰り返されるべきユニットは，*required_labels* をすべて含むはずである．このチェックによって，図 7 の 1 行目と 2 行目を表す原子ユニットはどちらも繰り返されるべきユニットではないことが分かる．もし，ユニットがこの条件を満たせば，そのユニットは繰り返されるべきユニットであるとみなされる．このため，図 7 の 3 行目を表す原子ユニットは繰り返されるべきユニットである．次にプロセスはこのユニットと同じ構造を持つユニットを探す．繰り返されるべきユニットを s とすると，s と同じ構造が繰り返し現れる部分を s+という新しい複合ユニットで置き換える．この複合ユニット s+は s の繰り返しを表す．表全体を表す複合ユニット *generalized_table* の中に置き換えが生じた場合，その新しい *generalized_table* はプロセスの新たな入力となる．これを繰り返し行い，我々は一般化された表構造，*generalized_repetitive_structure* を得ることができる．例えば図 7 の複合ユニットから図 8 の一般化構造が得られる．

また，このアルゴリズムの計算量と記憶量は，表形式の言語資源に含まれる行（横方向に繰り返される場合は列）の数を n とすると，$O(n)$になる．理論上は，1 つの部分配

列につき全てのユニットと比較するという処理を行うため，$O(n^2)$となってしまう．しかし，繰り返されるべきユニットを変更して，さらに繰り返しを発見するという処理の回数をあらかじめ定めた定数とすることで，$O(n)$となる．また，記憶量も $O(n)$となる．もし *generalized_table* に置き換えが生じた場合，新たなプロセスが新たな *generalized_table* とともに呼ばれ，「古い」プロセスの出力は「新しい」プロセスの出力となる．このため，プロセスは常に最新の *generalized_table* しか記憶する必要がないため，記憶量も $O(n)$となる．

2 つ目の課題はユーザによる例示の誤りである．例示の中に誤りがあると，表構造の一般化にも誤りが生じる．このため，ユーザには正確な例示を行うことが求められる．しかしながら，正しい一般化のためにどのような例示を行えばよいかは明確ではない．また，RDF の複雑さもユーザが例示を与える際の障壁となっている．そこで，全てのユーザが簡単に例示を与えることができるようにするため，我々は対話形式のシステムを開発した．このシステムでは，ユーザはシステムの表示する選択肢を選ぶだけで，適切な例示を行うことができる．まず，ユーザは例示を行いたいセルの座標を入力する．もしその座標が表の外側であるような場合は，ユーザは適切な座標を入力するよう求められる．次に，システムはそのセルに含まれている情報について，4 つのメタデータの候補を表示

する．候補は，翻訳元・翻訳先言語，対訳文，属性名，属性値の 4 つである．もしユーザが翻訳元・翻訳先言語や対訳文を選んだ場合，システムはユーザに言語コードを入力するように求める．言語コードとは各国の言語を 2 文字か 3 文字のアルファベットで記述したもので，たとえば英語であれば「en」である．属性名，属性値の場合は，「備考」や「索引」などの属性名を入力するように求める．これでひとつのセルに対する例示は終了である．ユーザは表に含まれるすべての種類のメタデータに対してこのような方法で例示を行う．この際に例示が制約を満たしているかのチャックも行い，1 つの対訳に同じ言語で記述された対訳文が 2 つ含まれているような場合は，例示をやり直すようユーザに求める．

本研究では，Web 上のコンテンツを自動的に解析し，大量に対訳を集めてくるのではなく，一つ一つの表に対してユーザが例示を与えなければならない．Web には膨大な量の有用な情報が蓄積されているため，本手法では人手による例示を与える作業も膨大となってしまう．しかしながら，機械翻訳などの言語サービスと連携できる Web サービスを表から作成することを考えると，やみくもに Web から大量の対訳を抽出する方法は精度の面でふさわしくない．むしろ精度の面では，有用な情報をユーザが見つけ出し，その情報から精度の良い一つの Web サービスを作成することの方が適している．そこで本研究では，すべてのユーザが容易に例示を行えるようにするため，このような対話式のシステムを開発した．

4. 評価

今回提案した手法でメタデータが生成できるかを確認するために，表形式の言語資源の持つ制約を利用したメタデータの半自動生成の有効性を評価した．評価に際して我々は Web 上に蓄積された言語資源からメタデータを生成するシステムを開発した [2]．評価を行った言語資源の総数は 50 であった．表 3 に評価を行った言語資源の例を示す．

まず，言語資源からメタデータを生成し，その精度について言語資源の持つ制約を利用する場合と利用しない場合についての比較を行った．また，結果を 4 つのカテゴリに分けた．カテゴリは，正しいメタデータが生成できる，メタデータの修正が必要である，誤ったメタデータが生成されている，メタデータが生成できない，の 4 つである．メタデータの修正が必要であるというカテゴリでは，英語の文に日本語で読み仮名がふってあるなど，不要なデータが含まれておりそれらを取り除く必要があった．誤ったメタデータが生成されているというカテゴリでは，英語の表現に対し，ja のような日本語を示すラベルがところどころ適用されてしまう場合などがあり，誤ったメタデータが出力されてしまった．表 4 にそれぞれのカテゴリに分類された言語資源の割合を示す．

表 4　一般化による表構造の解釈の精度

	制約を利用しない	制約を利用する
正しいメタデータが生成できる	6%	42%
メタデータの修正が必要である	8%	26%
誤ったメタデータが生成されている	42%	28%
メタデータが生成できない	44%	4%

表 3　評価を行った言語資源の例

言語資源名	URL	言語資源提供者
GIS 及び防災用語の多言語対応表	http://zgate.gsi.go.jp/2kokukan/termstable/tableex_ac_j.html	日中共同研究プロジェクト
化学用語英和辞典	http://www.geocities.jp/ryugaku_123/chem/chem-glossary.htm	個人ユーザ
文学・芸術英語用語集	http://www.rondely.com/zakkaya/dic7/lit1.htm#lit1	翻訳ワークショップ運営者
画像技術用語集	http://www.isj-imaging.org/word_collection/Word_Collection.html	日本画像学会
薬理学用語集	http://plaza.umin.ac.jp/JPS1927/glossary/ex-glossary-a.htm	日本薬理学会
カルテ略語辞典	http://www9.plala.or.jp/sophie_f/referene/dictionary/index.html	個人ユーザ
循環器学用語集	http://www.j-circ.or.jp/yougoshu/engine/searches/idx/	日本循環器学会
人間工学関連用語の対訳辞書	http://www.ergonomics.jp/jenc/jenc_dict.htm	日本人間工学会
心理学 専門用語集	http://www.educ.kyoto-u.ac.jp/cogpsy/personal/Kusumi/eng03term.htm	京都大学教育認知心理学講座
警察用語和英辞典	http://www.geocities.jp/ryugaku_123/eng/police-JPN-ENG.htm	個人ユーザ

次に，生成されたメタデータの再現率と適合率を計算し，制約を利用する場合としない場合とで比較した．表 5 に示すように，適合率も再現率も制約を利用した場合に上昇した．これは，制約を利用しない場合，一般化の誤りによって多くの誤ったメタデータが生成されてしまったためである．

表 5　生成したメタデータの適合率・再現率

	制約を利用しない	制約を利用する
適合率	0.41	0.85
再現率	0.42	0.87

言語資源の持つ制約を利用した一般化によって言語資源から対訳を抽出できたものに対して，そのコストを評価した．まず，例示に関して，言語資源の構造を解釈するために，何度ユーザが例示を与えることで言語資源から正しいメタデータが生成できるかを調べた．また，生成後のメタデータや対訳が言語サービスで全てそのまま使えた言語資源の割合を調べた．そのまま使えない対訳とは対訳中に不要な記述を含むようなもので，それらを取り除くための後処理が必要となる．この後処理の作業には新たなコストが発生するため，そのまま使えるかという観点からも評価を行った．結果は表 6 のようになった．

表 6　メタデータ生成のコスト

	生成したメタデータがそのまま使える	後処理が必要
各要素に対し 1 回の例示で抽出できる	43.8%	29.2%
各要素に対し複数回の例示が必要	16.7%	10.4%

既存の手法では，言語資源に例示を与える際に RDF による記述を行わなければならない．表を眺めただけで RDF による記述を行えるユーザでも，その記述を入力するコストは本手法で提案した例示にかかるコストと変わらない．なぜなら，対訳の構成要素がどのような関係であるかを一つ一つ入力する必要があるためである．また，このような例示に基づいて自動的に対訳を抽出する手法を用いずに，手作業で表から対訳をコピー&ペーストで抜き出す作業ともコストを比べるため，言語資源から抽出できた対訳の数と，修正しなければならなかった対訳の数を調べた．その結果，言語資源から得られたそのまま使える対訳の平均は，188.7 個で，手作業による言語サービスでそのまま使えなかった対訳の平均は，言語資源あたり，10.77 個であった．手作業で対訳を抽出する場合，本評価で用いた言語資源では一つ当たり 200 回程度のコピー&ペーストのコストが発生することになる．これに比べて，本手法を用いた抽出では，例示のための数回のコストと，修正のための 11 回程度のコストでメタデータと対訳を抽出できるという結果が得られた．

[2] 恣意性を取り除くため，http://www.kotoba.ne.jp からリンクを張られているすべての言語資源について評価を行った．

5. おわりに

本研究では Web 上に蓄積された表形式の言語資源からメタデータを半自動生成する手法について提案した．表形式の言語資源が持つ制約を利用することによって，我々は既存の一般化手法を適用することができた．本研究の主な貢献は以下の通りである．

まず，一般化誤りを大幅に減らすことができた． Web 上の 50 の言語資源を用いて行った本手法の評価実験では，正しく生成できたメタデータの適合率と再現率が 85%を超える結果が得られた．

次に，ユーザが一般化のために適切な例示を与えることを支援した．ユーザはシステムと対話することで簡単に例示が行える．ユーザはシステムが表示する選択肢を選ぶだけで，言語資源の制約に基づいた正しい一般化を行うための適切な例示を与えることができるようになった．

謝辞

本研究は，日本学術振興会科学研究費基盤研究(A)(21240014, 平成 21 年度～23 年度),京都大学グローバル COE プログラム「知識循環社会のための情報学教育研究拠点」から助成を受けた．

参考文献

[1] Ishida, T.: Language Grid: An Infrastructure for Intercultural Collaboration, *IEEE/ISPJ Symposium on Applications and the Internet (SAINT-06)*, pp. 96-100 (2006).

[2] Chen, H., Tsai, S. and Tsai, J.: Mining Tables from Large Scale HTML Texts, *18th International Conference Computational Linguistics (COLING2000)*, pp. 166-172 (2000).

[3] Wang, H., Wu, S., Wang, I., Sung, C., Hsu, W. and Shih, W.: Semantic search on Internet Tabular Information Extraction for Answering Queries, *9th International Conference on Information and Knowledge Management (CIKM2000)*, pp. 243-249 (2000).

[4] Embley, D., Tao, C. and Liddle, S.: Automatically Extracting Ontologically Specified Data from HTML Tables with Unknown Structure, *21st International Conference Conceptual Modeling (ER2002)*, pp. 322-337 (2002).

[5] Tijerino, Y., Embley, D., Lonsdale, D. and Nagy, G.: Ontology Generation from Tables, *4th International Conference on Web Information Systems Engineering (WISE2003)*, pp. 242-252 (2003).

[6] Hurst, M.: Layout and Language: Beyond Simple Test for Information Interaction-Modeling the Table, *2nd International Conference on Multimodal Interfaces (ICMI-99)*, pp. 243-249 (1999).

[7] Pivk, A., Cimiano, P. and Sure, Y.: From Tables to Frames, *3rd International Semantic Web Conference (ISWC-04)*, pp. 166-181 (2004).

[8] Tanaka, M. and Ishida, T.: Ontology Extraction from Tables on the Web, *IEEE/IPSJ Symposium on Applications and the Internet (SAINT-06)*, pp. 284-290 (2006).