アルファベット表記とカタカナ表記の対応規則の生成

尾上 徹[†] 梅村 恭司[††] 岡部 正幸[†††]

日本人は外国語を表記する際，カタカナ表記に置き換えることが多い．人名の場合には，ある表記に対して複数のカタカナ表記があるため，その複数のカタカナ表記全てを網羅的に列挙する辞書を作成することは合理的ではない．そこで，我々はアルファベット表記とカタカナ表記を対応させた規則の自動生成を行なった．我々は，もっともらしい分割点を見つける手法により規則の抽出を行い，その得られた規則を長い規則と短い規則に分け，長い規則を短い規則で置換することにより新たな規則の生成を行なった．この拡張した規則について評価を行なったところ，検索性能が向上したことを報告する．なおこのとき，アルファベット表記の言語の母音を示す文字以上の知識は用いていない．

Generating Rules Between Katakana Notation and its Alphabet Notation

Toru Onoe,[†] Kyoji Umemura[††] and Masayuki Okabe[†††]

Generally, Japanese often replaces alphabet notation with correspond katakana notation when they write a foreign language. A personal name has plural katakana notation. Therefore, it is unreasonable to make a dictionary containing the plural katakana notation exhaustively. In this report, we have generated a correspondence rule of katakana notation and alphabet notation automatically. We have extracted the rule by a certain method that finds a plausible divide in the notation; divide the provided rule into rule of long and short character length rule; generate new rule by substitute the long one with the short one. Our evaluation results show that our extended rule gives higher performance than original rule. Our method uses little knowledge such as the list of letters corresponding to vowels.

1. はじめに

日本人は外国語を表記する際，カタカナ表記に置き換えることが多い．外国語で記述された情報から検索を行う際，調べたい事柄のカタカナ表記だけ知っていたとしても元の綴りを知らなければ検索することは難しい．特に人名の場合には，ある表記に対して複数のカタカナ表記があり，その複数のカタカナ表記全てを網羅的に列挙する辞書を作成することは合理的ではないと考えられる．

そこで，これを解決する一つの手段としてアルファベット表記とカタカナ表記を対応させた規則（以下単に規則）を自動生成することを考える．この規則は外国語のアルファベット表記とカタカナ表記が対応したデータの集合から自動的に取り出すこととする．このとき，アルファベット表記の言語の母音を示す文字以上の知識は用いない．人名の発音記号が入手できるとすればアルファベット表記を用いるよりも正確なシステムが作れると思われるが人名の発音記号を入手するのはコストが高いと考える．よって，本研究では発音記号を使わないこととした．

カタカナによる英単語の検索はこれまでにも行なわれている．例えば，宮内[1)] は英単語の発音記号からカタカナ表記を作り，検索するカタカナの表記ゆれを変換表で解消し，検索を行なった．しかし，この方法は規則の抽出に人手を用いており，発音記号の情報を用いている．後藤ら[2)] の手法は，文脈情報と英語のWWW 文書，英語・日本語からの発音記号への変換規則を用いて固有名詞の検索を行なう．これは，文脈情報を用いて検索精度を高め，WWW 文書を用いることで最新の固有名詞を検索できる点で有用であるが，これもまた発音記号への変換規則の作成に人手を用いている．また，辞書の発音記号を直接用いない検索方法[3)] もあるが，この方法もルールの抽出を人手により

† 豊橋技術科学大学情報工学系
Information and Computer Science, Toyohashi University of Technology
†† 豊橋技術科学大学情報知能工学系
Computer Science and Engineering, Toyohashi University of Technology
††† 豊橋技術科学大学情報メディア基盤センター
Information and Media Center, Toyohashi University of Technology

行なうため，規則の変化に柔軟ではない．

発音記号を直接用いずに読みを得る方法として 住吉ら[4] があるが，これも英文字列を変換する変換テーブルを人手により作成する必要がある．読まれうるカタカナの表記の揺らぎを解消する飯田ら[5]，伍井ら[6] の手法もまた，人手によるルールの作成が必要であり，自動的に規則を得ることができない．

翻訳を利用し英語文献に対する日本語検索を行なう方法として藤井ら[7] の手法があり，これは固有名詞以外の検索も行なえる点で有用である．しかし，人名・固有名詞は日本語表記が一意に定まらず，辞書に含まれない可能性がある．このため，藤井ら[7] の手法を我々の検索対象である人名に対して用いることは難しい．

外国語の情報から，カタカナ表記で人名の検索を行う際に重要となることは，その検索者が意図したものが含まれるならそれにヒットすることである．このため，入力したカタカナに一つのアルファベットを対応させる必要はなく，それに対応する候補の集合の中に意図したものが含まれていればよいと考えることができる．よって，本研究では入力に対するマッチングの候補に正解が含まれていることを規則の検索性能とした．そして，カタカナ表記とアルファベット表記を一意に結びつけることは目的とはしないこととした．本研究を実際の検索システムに用いる場合，検索結果に対して，結果を絞り込む何かしらの処理（例えばユーザによる絞込みなど）が加えられることを想定している．

この立場から，検索効率を極端に低下させないためにもっともらしさをある水準以上持ち，かつ，もれなく検索するためできる限り多くの規則を得ることを我々は重視する．ゆえに，KNIGHT[8] とは問題設定が異なる．

カタカナによるアルファベットの人名検索は，アルファベット文字列を規則によりカタカナに置換し，その結果が入力のカタカナと一致するかどうかを調べる作業といえる．置換は何度も行われるため，できる限り単純であることが望まれる．よって本研究では単純な置換による検索でも効果を発揮する規則を生成することを目的とした．

また，本研究は現在の計算機で実現できるかどうかは考慮するが，計算不可を少なくすることは目的としていない．このため，本研究は現在の計算機で実現できる範囲で実験を行い検討した．

人名事典からのアルファベット表記に対応するカタカナ表記の規則の抽出は，増田ら[9] により提案されている．本研究では，増田らの手法[9] をベースに抽出された規則の改良について扱う．なお規則の評価は，増田らの用いた規則の評価尺度（綴りの復元率，読みの復元率）と，新たに提案する規則の評価尺度（逆綴り復元率，逆読み復元率）によって行なうこととした．その結果，増田らの用いた規則の評価尺度において規則の性能が有意に向上したことを報告する．そして，なぜそのような結果となったかについても考察を行なう．

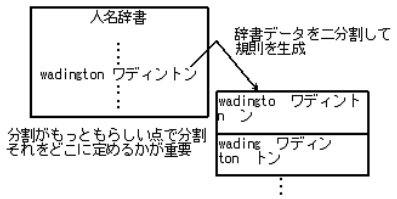

図 1 人名辞書からの対応規則の抽出例

2. 分割点を発見することによる対応規則抽出方法

我々がベースとして用いる増田らの手法[9] は，分割点を発見することにより対応規則を抽出する方法であるといえる．この手法は，**図 1** のように，人名辞書データを入力とし，アルファベット綴りとそれに対応する読みをそれぞれ二分割することで，アルファベットの綴りに対応する日本語読みの規則（以降，対応規則もしくは単に規則と表記）を生成する方法である．

本手法により求める規則は，アルファベットを対応するカタカナに変換するために使うのではなく，カタカナ表記の単語に対し，検索の対象となりうる範囲を定めるものである．このため，本手法では検索効率を極端に低下させないために，もっともらしさをある水準以上持つ規則を抽出し，また，もれなく検索するために，できる限り多くの規則を得ることを重視する．

2.1 記号の定義

ここでは，本章にて用いる記号の定義を行なう．データ（人物名の綴りとその読みの対）の集合を D で表し，$\mathrm{D} = \{d_1, \cdots, dh, \cdots, d_l\}$ とする．ここで l はデータ数を表し，d_h は $d_h = (A_h, K_h)$ と表す．A_h はアルファベット文字列，K_h はカタカナ文字列で，$A_h = a_1 \cdots a_m$，$K_h = k_1 \cdots k_n$ と表す．ここで，$a_i (1 \leq i \leq m)$ はアルファベット文字，$k_j (1 \leq j \leq n)$ はカタカナ文字である．

データを $d_h = (A_h, K_h)$ の A_h を語頭と語尾の 2 つに分割する．同様に K_h も語頭と語尾の 2 つに分割する．A_h の語頭と K_h の語頭，A_h の語尾と K_h の語尾で組を作る．このそれぞれの組は対応規則になる可能性がある，対応規則の候補である．ここで，d_h の語頭

対応規則候補を $cfak_h^{ij}$，語尾対応規則候補を $crak_h^{ij}$ とする．これは，次式のように表すことができる．

$$\begin{aligned} cfak_h^{ij} &= (a_1 \cdots a_i, k_1 \cdots k_j) \\ crak_h^{ij} &= (a_{i+1} \cdots a_m, k_{j+1} \cdots k_n) \\ & (1 \le i \le m-1, 1 \le j \le n-1) \end{aligned} \quad (1)$$

この対応規則候補は，語尾語頭を区切る場所によりそれぞれ全部で $(m-1)(n-1)$ 個作ることができる．ここで，$cfak_h^{ij}$ の集合を CFAK_h，$cfak_h^{ij}$ の集合を CRAK_h と表す．なお，この集合はそれぞれ $(m-1)(n-1)$ 個の対応規則候補を持つ．

データ集合 D の任意の d_h から作られる対応規則候補の集合 CFAK_h と CRAK_h が D において出現する頻度をそれぞれ調べる．$cfak_h^{ij}$ の出現頻度を $\mathrm{f}(cfak_h^{ij})$，$crak_h^{ij}$ の出現頻度を $\mathrm{f}(crak_h^{ij})$ と表し，それぞれ i, j について表にしたものを $\mathrm{F}(cfak_h)$，$\mathrm{F}(crak_h)$ と表す．この表を出現頻度表と呼ぶこととする．**表 1** に出現頻度表 $\mathrm{F}(cfak_h)$ と $\mathrm{F}(crak_h)$ を示す．

2.2 対応規則抽出法

本手法は，外国語に依存した知識を用いずに対応規則をデータ集合 D から抽出する．扱うデータには英語のみではなくドイツ語，フランス語等複数の言語が含まれ，読み仮名は全て日本語（カタカナ）である．そこで，本手法では次の 2 つの日本語の知識を用いる．

(1) 母音はカタカナ表記の区切り（変音記号のついた母音も母音とする）

(2) 促音（ッ）と長音（ー）は語頭に現れない

図 2 に，これらの知識を用いて D から対応規則を抽出するためのアルゴリズムを示す．ここで，T は閾値，R は信頼限度というパラメータである．このパラメータについては次節にて述べる．

本手法は分割点を 1 つアルファベット文字列とその読みのカタカナ文字列に定めることでルールを得る方法であるから，アルゴリズムは妥当とみなされる語頭規則もしくは語尾規則が抽出されたら，その残りの文字列もまた規則として取り出すようになっている．また，この手法では，アルファベット文字列が 1 文字もしくはカタカナ文字列が 1 音節の場合，分割できないため規則は生成されない．

2.3 パラメータの実験値

閾値 まず，外国語のアルファベット表記はアルファベットの前後関係によらないというモデル，すなわち子音の次に母音が出現するかどうかは独立であるというモデルを仮定する．このモデルのデータ集合から出現頻度表を作成すると，$\mathrm{F}(\mathrm{CFAK}_h)$ では $\mathrm{f}(cfak_h^{ij})$ は i, j について単調減少し，$\mathrm{F}(\mathrm{CRAK}_h)$ では $\mathrm{f}(crak_h^{ij})$ は i, j について単調増加することが予測できる．しかし，実際のデータでは出現頻度表の数の並びは単調にはならない．注目するカタカナ文字列に対してアルファベット文字列が実際に対応する可能性がなくなったとき，その文字列対の出現頻度は上記に仮定したモデルにおける頻度より急激に減少する．この急激に変化する点を決めるのが閾値である．増田らは閾値として $1/3$ が妥当であるとした．

```
//出現頻度表 F(cfak_h) からの規則の抽出
for(j=1; j<=n-1; j++)
  for(i=1; i<=m-1; i++)
    if(R>f(cfak_h^ij)) break;
    if(a_i は母音ではない) continue;
    if(T>f(cfak_h^(i+1)j)/f(cfak_h^ij)){
      cfak_h^ij を語頭対応規則として抽出;
      crak_h^ij を語尾対応規則として抽出;
      break;
    }
  }
}

//出現頻度表 F(crak_h) からの規則の抽出
for(j=n-1; j>=1; j–)
  for(i=m-1; i>=1; i–)
    if(R>f(crak_h^ij)) break;
    if(a_i は母音である) continue;
    if(T>f(crak_h^(i−1)j)/f(crak_h^ij)){
      cfak_h^ij を語頭対応規則として抽出;
      crak_h^ij を語尾対応規則として抽出;
      break;
    }
  }
}
```

図 2 分割点を発見することによる対応規則抽出方法のアルゴリズム

信頼限度 使用するデータにはノイズが存在することが多いが，それを全て取り除くことは困難である．そこで，出現頻度が低いものをノイズとして，対応規則を得る過程から取り除くことを考える．この信頼する出現頻度数を定めるのが信頼限度で，信頼限度以上の出現頻度数ならばそのデータは信頼できると考える．増田らは信頼限度として 10 が妥当であるとした．この数値は対象により妥当な数値を定めるべきであるが，本実験では，増田らと同様のコーパスを用い，条件をそろえて比較を行なうために同様の値を用いることとした．

表 1 出現頻度表 $\mathrm{F}(cfak_h)$ と $\mathrm{F}(crak_h)$

	k_1	k_2	$\cdots$	k_{n-1}	k_n
a_1	$\mathrm{f}(cfak_h^{11})$	$\mathrm{f}(cfak_h^{12})$	$\cdots$	$\mathrm{f}(cfak_h^{1n-1})$	$\mathrm{f}(cfak_h^{1n})$
a_2	$\mathrm{f}(cfak_h^{21})$	$\mathrm{f}(cfak_h^{22})$	$\cdots$	$\mathrm{f}(cfak_h^{2n-1})$	$\mathrm{f}(cfak_h^{2n})$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
a_{m-1}	$\mathrm{f}(cfak_h^{m-11})$	$\mathrm{f}(cfak_h^{m-12})$	$\cdots$	$\mathrm{f}(cfak_h^{m-1n-1})$	$\mathrm{f}(cfak_h^{m-1n})$
a_m	$\mathrm{f}(cfak_h^{m1})$	$\mathrm{f}(cfak_h^{m2})$	$\cdots$	$\mathrm{f}(cfak_h^{mn-1})$	$\mathrm{f}(cfak_h^{mn})$

	k_1	k_2	$\cdots$	k_{n-1}	k_n
a_1	$\mathrm{f}(cfak_h^{11})$	$\mathrm{f}(cfak_h^{12})$	$\cdots$	$\mathrm{f}(cfak_h^{1n-1})$	$\mathrm{f}(cfak_h^{1n})$
a_2	$\mathrm{f}(cfak_h^{21})$	$\mathrm{f}(cfak_h^{22})$	$\cdots$	$\mathrm{f}(cfak_h^{2n-1})$	$\mathrm{f}(cfak_h^{2n})$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
a_{m-1}	$\mathrm{f}(cfak_h^{m-11})$	$\mathrm{f}(cfak_h^{m-12})$	$\cdots$	$\mathrm{f}(cfak_h^{m-1n-1})$	$\mathrm{f}(cfak_h^{m-1n})$
a_m	$\mathrm{f}(cfak_h^{m1})$	$\mathrm{f}(cfak_h^{m2})$	$\cdots$	$\mathrm{f}(cfak_h^{mn-1})$	$\mathrm{f}(cfak_h^{mn})$

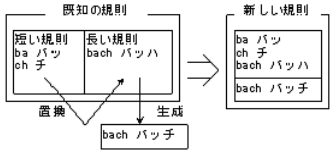

図 3 置換による新しい規則の生成

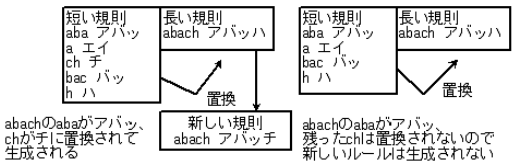

図 4 人名辞書からの対応規則の抽出例

3. 置換による対応規則抽出法

我々の対応規則抽出法は，増田らの分割点を発見することによる対応規則抽出方法により得られた対応規則を用いて，そこから新たな対応規則を生成するものである．その方法は，既知の対応規則を長いものと短いものに分け，短い対応規則で長い対応規則を置換して新たな規則を作り，それを既存の規則に追加するというものである．**図 3** に新しい規則の生成例を示す．この例では ba バッ, ch チ を短い規則，bach バッハ を長い規則と考え，bach を置換することで bach バッチ という新しい規則が生成され，これまでの規則に追加される．このように，置換することにより規則は多く生成されるが，この規則の中に検索に役立つものが含まれていればよいという立場を我々はとる．

3.1 記号の定義

本節以降用いる記号の定義を**表 2** に示す．

3.2 手　　順

まず，人名辞書などのアルファベット表記とカタカナ表記が対となったデータから，分割点を発見することによる対応規則抽出方法により規則（R）の抽出を行なう．R は RF と RR からなる．次に，この RF と RR をさらに短い規則と長い規則に分ける．長い規則，短い規則というのは，カタカナの文字数やアルファベットの文字数に基づいて定める．具体的な文字数については，R がどの程度の文字数の対で構成されているかを調べ，それに基づいて主観的に決定することとした．本研究では，カタカナ，アルファベット両方の場合について行い，規則の性能を調べることとした．そして，得られた短い規則で，長い規則を置換することで，新しい対応規則を得る（カタカナ文字数準拠の場合は RRK，アルファベット文字数準拠の場合は RRA）．

3.3 置 換 方 法

長い規則を短い規則で置き換えるときの置換方法は，先頭からの最長一致で行なう．このとき，バックトラックなどのような手法は用いず，先頭から最も長く置換できるものを当てはめ，当てはめるものがなくなるもしくは，全て当てはめが完了した場合に終了する．次にこの例を示す．**図 4**(a) では，長い規則 abach を短い規則 aba, ch で置換することで abach アバッチが生成される．(b) では，最も長い規則 aba を当てはめた結果，残りを当てはめることができず，バックトラックを用いないため，規則は新たに作成されない．

4. 実　　験

4.1 実 験 方 法

4.1.1 規則の抽出及び評価に使用するデータ

本研究では，カタカナとアルファベットの対応規則を抽出する対象として，外国人名のアルファベット表

表 2　記号の定義

R	(Rule)	分割点を発見することによる対応規則抽出方法により抽出された対応規則の集合
RF	(Rule Front)	R に含まれる語頭対応規則対の集合
RR	(Rule Rear)	R に含まれる語尾対応規則対の集合
RFSK	(Rule Front Short Katakana)	カタカナ１～３文字の短い語頭対応規則
RFLK	(Rule Front Long Katakana)	カタカナ４～９文字の長い語頭対応規則
RRSK	(Rule Rear Short Katakana)	カタカナ１～３文字の短い語尾対応規則
RRLK	(Rule Rear Long Katakana)	カタカナ４～９文字の長い語尾対応規則
RFSA	(Rule Front Short Alphabet)	アルファベット１～３文字の短い語頭対応規則
RFLA	(Rule Front Long Alphabet)	アルファベット４～１５文字の長い語頭対応規則
RRSA	(Rule Rear Short Alphabet)	アルファベット１～３文字の短い語尾対応規則
RRLA	(Rule Rear Long Alphabet)	アルファベット４～１５文字の長い語尾対応規則
RRK	(Rule Replace Katakana)	カタカナに基づき長い規則を短い規則で置き換えた対応規則の集合
RRA	(Rule Replace Alphabet)	アルファベットに基づき長い規則を短い規則で置き換えた対応規則の集合

記とカタカナ表記が対応した言語混合の人名事典[10]より成形したデータ）を用いることとした．この人名事典は 31847 個の人物名の対応データからなる．このうち 24000 個をランダムに取り出し，これをテストデータとする．そして，残り 7847 個を学習データ（対応規則抽出の対象）とする．この人名事典からランダムに生成する，テストデータ（対応データ 24000 個）と学習データ（対応データ 7847 個）の組をデータセットと呼ぶこととする．本実験では，このデータセットを 10 個構成し，それぞれについて規則の抽出を行い，規則の性能の比較を行なう．規則の抽出については 4.1.2 節，評価方法については 4.2 節で述べる．

4.1.2　規則の抽出

まず，我々の規則抽出に用いるための対応規則を，増田らの手法（分割点を発見することによる対応規則抽出方法）[9] を用いて学習データから抽出することで作成する．ここで用いる具体的なパラメータは 2.3 節にて述べたものを用いる．

この得られた対応規則 R（RF, RR）の内，短い対応規則を用いて長い対応規則を置き換える．置き換えは 3.2 節で述べたように先頭からの最長一致で置き換える．対応規則は,「アルファベット表記」と「カタカナ表記」の組であるから，アルファベット表記に基づく置き換えとカタカナ表記に基づく置き換えの 2 通りの実験を行なう．すなわち，カタカナの長い規則（RFLK, RRLK）をカタカナの短い規則（RFSK，RRSK）で置き換えて対応規則の集合 RRK を，アルファベットの長い規則（RFLA，RRLA）をアルファベットの短い規則（RFSA，RRSA）で置き換えて対応規則の集合 RRA を得る．

ここで，対応規則をカタカナ文字数別に分けたところ 1〜9 文字の対応規則が，アルファベット文字数別に分けたところ 1〜15 文字の対応規則があることがわかった．

そこで，短い対応規則として１～３文字の対応規則を考え，残りの対応規則を長い対応規則として置き換えの対象とすることとした．なお，この１～３文字という数字は主観的に定めたものであり，明確な根拠はない．この数字を変化させた場合の結果について検討することは興味深いものの，今実験では１～３文字を短い規則とする場合のみについて検討することとした．

以上のようにして得られた対応規則集合 R，RRK，RRA それぞれを用いた場合のテストデータ（24000）に対する綴りの復元率と読みの復元率（後述する）を求め，これによりそれぞれの評価を行なう．

4.2　評 価 方 法

本実験では，増田らの用いた綴りの復元率と読みの復元率という評価尺度と，新たに定める逆綴り復元率と逆読み復元率という４つの評価尺度で対応規則の性能を測ることとする．

綴りの復元率 綴りの復元率とは，テストデータの各アルファベット表記を，対応規則集合を用いてカタカナに変換できる（読みの候補を作ることができる）割合である．これは，十分な量の規則が生成できているかを測る．ただし，置き換えられた読みが，元のデータと同じか異なるかは考えない．対応規則の当てはめは先頭から最長一致で当てはめる単純な方法で行った．

読みの復元率 読みの復元率とは，綴りを復元できたデータの集合において，読みが正しかった割合を表す．この比率は規則が生成できた場合に，元データにある正解のカタカナ表記で目的のデータが発見できる確率に相当する．これにより規則の性能，すなわち，実際のデータベース検索における検索可能な確率を測る．読みが正しいとは，アルファベット文字列の読み方を規則から生成し，その生成された読み方の集合に正解（辞書に記された読み方）が含まれている場合である．ただし，辞書に複数とおりの読み方がある（例えば adrian に対してアドリアン，エイドリアンとい

う読み方がある）場合，それらは別々のデータとして扱われるため，adrian-アドリアンのペアに対してエイドリアンという読みしか生成できなければ，そのペアについて不正解として扱われる．

逆綴り復元率 逆綴り復元率とは，テストデータの各カタカナ表記を，対応規則集合を用いてアルファベットに変換できる割合である．綴りの復元率と同様に，置き換えて得られたアルファベット表記が元のデータと同じか異なるかは考えない．また，規則の適応も先頭からの最長一致で同様に行なう．

逆読み復元率 逆読み復元率とは，カタカナをアルファベット表記に置き換えられたものの内，その綴りが元のデータ（正解）と等しかったものの割合である．新たにこれを評価尺度として設けたのは，増田らの評価尺度である読みの復元率は，アルファベット表記でカタカナ表記を検索するシステムに用いる規則集合の評価に相当し，カタカナ表記でアルファベット表記を検索するシステムに用いる規則集合の評価には十分ではないのではないかと考えたためである．

5. 実験結果

実験の結果得られた語頭対応規則集合，語尾対応規則集合の大きさは**表 3** のようになった．綴りの復元率と読みの復元率をそれぞれの対応規則について求めた結果を**表 4** に示す．逆綴り復元率，逆読み復元率をそれぞれの対応規則について求めた結果を**表 5** に示す．

本実験では，10 個のデータセット全てにおいて，規則集合 RRK， 規則集合 RRA の両方の綴りの復元率は，元の規則集合 R と変わらない結果となり，読みの復元率は，10 個のデータセット全てにおいて，R よりも RRK が優れており，RRK よりも RRA が優れているという結果となった．読みの復元率の平均値は R が 0.2856 で RRK は 0.2952 と，R より 0.0096 だけ高く，RRA は 0.3118 と，R より 0.0262 だけ高い．逆読みの復元率は，全てのデータセットにおいて R が最も高く，RRA が最も低いという結果となった．

これらのことから，長い規則を短い規則で置換することで，新たな対応規則を作ることは，増田らの評価尺度である読みの復元率によると検索可能性の向上に有効であることがわかる．しかしながら，今回新たに設けた評価尺度である逆読み復元率では検索可能性の低下が起こるという評価となった．このように，用いる評価尺度により規則の検索性能の評価の優劣が変わるという結果が得られた．6 章にて，なぜこのような結果となったか，その原因について考察し，これを解決することで，どちらの評価尺度でも置換により対応規則を増やす我々の手法により得られる対応規則を用いることで検索の性能が向上することを述べる．

表 3 対応規則集合の大きさの比較

	R		RRK		RRA	
	語頭	語尾	語頭	語尾	語頭	語尾
1	6814	5403	96925	122759	224072	270100
2	6668	5364	165016	96334	302451	202113
3	6772	5369	126468	157326	233087	354259
4	6693	5314	150255	148705	319749	298840
5	6865	5347	236383	175605	469564	506607
6	6808	5432	160190	107475	273370	235320
7	6785	5414	190242	218303	383751	489399
8	6785	5414	190242	218303	383751	489399
9	6538	5303	129878	127008	266415	373258
10	6648	5426	169666	116494	294904	243361

表 4 綴りの復元率と読みの復元率

	綴りの復元率			読みの復元率		
	R	RRK	RRA	R	RRK	RRA
1	0.9469	0.9469	0.9469	0.2808	0.2914	0.3070
2	0.9448	0.9448	0.9448	0.2864	0.2964	0.3120
3	0.9527	0.9527	0.9527	0.2844	0.2950	0.3087
4	0.9494	0.9494	0.9494	0.2793	0.2868	0.3044
5	0.9459	0.9459	0.9459	0.2807	0.2908	0.3061
6	0.9518	0.9518	0.9518	0.2811	0.2885	0.3060
7	0.9506	0.9506	0.9506	0.2897	0.2995	0.3160
8	0.9478	0.9478	0.9478	0.2833	0.2942	0.3119
9	0.9486	0.9486	0.9486	0.2992	0.3079	0.3263
10	0.9524	0.9524	0.9524	0.2906	0.3016	0.3200
平均	0.9491	0.9491	0.9491	0.2856	0.2952	0.3118

表 5 逆綴り復元率と逆読み復元率

	逆綴り復元率			逆読み復元率		
	R	RRK	RRA	R	RRK	RRA
1	0.9058	0.9055	0.9050	0.2612	0.2445	0.2141
2	0.8932	0.8928	0.8927	0.2590	0.2387	0.2037
3	0.9004	0.9002	0.8999	0.2668	0.2441	0.2140
4	0.9003	0.9000	0.8996	0.2552	0.2365	0.2069
5	0.9007	0.9006	0.9001	0.2541	0.2308	0.2029
6	0.9039	0.9037	0.9035	0.2618	0.2432	0.2097
7	0.9019	0.9019	0.9020	0.2631	0.2467	0.2098
8	0.8965	0.8965	0.8961	0.2650	0.2454	0.2131
9	0.8992	0.8988	0.8987	0.2676	0.2473	0.2106
10	0.9154	0.9152	0.9148	0.2554	0.2386	0.2039
平均	0.9017	0.9015	0.9012	0.2609	0.2416	0.2089

6. 考　　察

表 3 をみると，規則を用いた置き換えにより対応規則集合の大きさは，10 個全てのデータセットにおいて RRA は最も大きく，R は最も小さな値をとった．我々の置換による規則の抽出は，アルファベット文字列を既存の規則でアルファベットからカタカナ文字列へと置換して新しい規則を得るため，アルファベット

表 6 集合の大きさの例

対応規則		大きさ
ja	ヤ，ヤン，ジャ	3
g	ク，グ	2

に基づき長さを決めたほうが，あるアルファベット文字列を置換できる可能性が増加し，それに対する読み（規則）が多く生成されると考えられる．このため表 3 のような結果となったと考えられる．

ここで，対応規則集合の大きさとは，アルファベットに対応するカタカナの個数である．例えば，対応規則集合が**表 6** のようになっていたとするならば，この対応規則集合の大きさは 5 である．

例えば R では，”wadingto”というアルファベットに対してボディントという 1 つの規則しか割り当てられなかった．しかし，RRK では短い規則で置換することで，”wadingto”に**表 7** に示す 40 個の規則が割り当てられた．このうち，例えばワディグトやワディント，ワディンクト，ワディングトなどは，”wadhingto”が英語のスペルの一部であると考えた場合，実際に人がそのように読みうる可能性があると考えた．

このように，置き換えることにより，膨大な量の読み方が生成されるが，その中には前述のように人が読んでもおかしくない（そのように読まれる可能性があると思われる）ものが含まれているため，検索に全く役に立たない規則が増えたわけではないと考えることができる．

実験結果の表 4 をみると，全 10 個のデータセットについて規則の抽出，性能の比較をそれぞれ行なったところ，平均的に RRA の読みの復元率が最もよく，R の読みの復元率が最も悪い結果となったことがわかる．10 個中 10 個のデータセットにおいて RRA は R, RRK よりも高い性能（読みの復元率）を示した．この結果について符号検定を行い，RRA の性能と，R 及び RRK の読みの復元率による性能の間に有意差があるかどうかを考える．まず，帰無仮説 H_0 と対立仮説 H_1 を以下のように定める．

H_0： RRA と，R 及び RRK の読みの復元率の間に有意差はない

H_1： RRA と，R 及び RRK の読みの復元率の間に有意差がある

これらの規則の読みの復元率の分布が等しいという仮定の下で R 及び RRK の読みの復元率が RRA の読みの復元率を一度も上回らない確率は，

$$\frac{1}{2^{10}}\left({}_{20}C_0\right) = 0.0009766\cdots < 0.001 \qquad (2)$$

となるため，危険率 1%で帰無仮説 H_0 は棄却され，対立仮説 H_1 が採択される．このことから RRA の読みの復元率と R 及び RRK の読みの復元率の間に有意な差があると考えることができる．また，R と RRK の読みの復元率の間にも有意な差があることは同様の検定を行なうことで分かる．

このように，対応規則の増加に伴い読みの復元率が実際に向上したことから，すでに規則として取り出された規則で置換することにより生成された規則が有効に作用したことが分かる．

先のように読みの復元率は置換により改良されたが，表 5 を見ると，逆読み復元率は置換により低下することが分かる．R よりも RRK, RRA は低く，RRK よりも RRA は低い値となった．

このような結果となった理由について，データセット 1 における実際の例を用いて考察する．データセット 1 において，規則集合 R で置換することで正解の綴りを作れたものはテストデータ 24000 個の内 5679 個で，規則集合 RRA で置換することで正解の綴りを作れたものは 4650 個であった．このうち，R で正解を作れ，RRA では正解を作れなかったデータの数は，1291 個で，R で正解を作れず，RRA で正解を作れたデータの数は 262 個であることがわかった．このことから，置換により得られた新たな規則で拡張したことにより，正解の綴りに置換できるようになる場合と，正解の綴りに置換できなくなってしまう場合の両方があるということがわかる．

まず，**図 5** に R で正解の綴りに置換でき，RRA では正解の綴りに置換できなかった場合の実例を示す．ここでは，”エステルグレン”というデータに対して対応規則を用いてアルファベット綴りへの先頭からの最長一致で置換を行なっている．R では，まず”エステルグレン”の”エステル”に対して，次に”グレ”に対して，最後に”ン”に対して置換が行なわれる．この結果生成される綴りの集合の中に正解である”östergren”が含まれるので，この置換は成功であり，逆読み復元率は増加する．RRA では，まず”エステルグレン”の”エステル”に対して，次に”グレン”に対して置換が行なわれ終了する．結果として得られる綴りの集合の中に正解である”östergren”は含まれないため，この置換は失敗となり，逆読み復元率は増加しない．この RRA の”グレン gran”の対応規則は**図 6** のようにして対応規則集合に追加されたものである．この対応規則が追加されたことで，”グレン”を”グレ”と”ン”に分けることで正しく”gren”に変換できるにもかかわらず，最長一致での変換であるために規則の適応が妨げられている．このことが RRA での逆読み復元率の低下につな

表 7 wadingto の対応規則の比較

R	ボディント			
RRK	ウディグト	ワディグト	ウォディグト	ウェイディグト
	ウディグリット	ワディグリット	ウォディグリット	ウェイディグリット
	ウディント	ワディント	ウォディント	ウェイディント
	ウディンリット	ワディンリット	ウォディンリット	ウェイディンリット
	ウディンクト	ワディンクト	ウォディンクト	ウェイディンクト
	ウディンクリット	ワディンクリット	ウォディンクリット	ウェイディンクリット
	ウディングト	ワディングト	ウォディングト	ウェイディングト
	ウディングリット	ワディングリット	ウォディングリット	ウェイディングリット
	ウディリングト	ワディリングト	ウォディリングト	ウェイディリングト
	ウディリングリット	ワディリングリット	ウォディリングリット	ウェイディリングリット

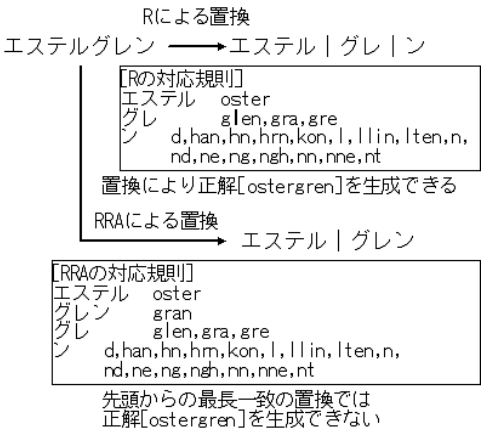

図 5 R と RRA による置換例 1

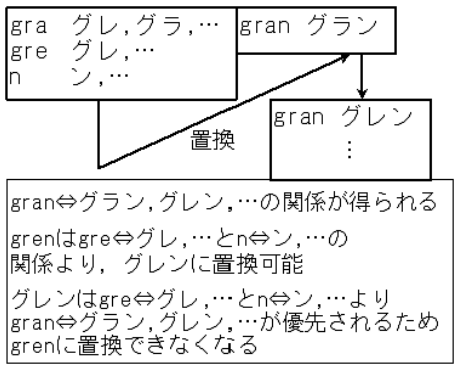

図 6 置換による規則の追加と影響

がった．

しかしながら，置換により規則が追加されることで，逆読み復元率の向上につながった実例もある．これを**図 7** に示す．これは”アルドリッチ” というデータに対して対応規則を用いてアルファベット綴りへの先頭からの最長一致で置換を行なっている．R では正解の

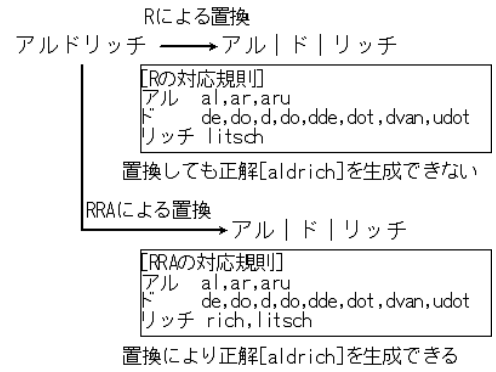

図 7 R と RRA による置換例 2

綴りを作れず，RRA では正解の綴りを作ることができた．この理由は，RRA が”リッチ”と”rich”の対応規則を持っていたためである．これは置換により追加された対応規則であり，R にはこれがないため正解を導けなかった．

データセット 1 において図 5 と同様のパターンで RRA の逆読み復元率の低下につながったのが 1291 個，図 7 と同様のパターンで RRA の逆読み復元率の増加につながったのが 262 個あり，今実験では低下につながる場合のほうが多かったため結果として逆読み復元率の低下が起こったと考えられる．このことから，図 5 のパターンの置換ミスを解決することで逆綴り復元率を向上させることができるといえる．

表 7 より，規則の置き換えにより，人間の読みと一致する可能性がある規則が生成されることが分かる．規則の置き換えにより，正解に貢献した例を以下に示す．**表 8** は抽出した規則で wadington（ワディントン）という人名に読みの候補を生成した結果である．表 8 において，R は 1 × 27=27 通りの読みを生成したが，正解を生成することはできず，RRK は 41 × 27=1107 通りの読みを生成し，正解の読みを生成す

ることができた．このように，新しい規則が作られることで読みの復元率が向上したことがわかる．また，RRA ではより多くの規則を置き換え，正解の候補を増加させたため，性能（読みの復元率）がより向上したことが推測できる．

今回用いた人名辞書には wadington の読みはワディントンのみしか含まれていなかったが，それ以外にも人がそのように読みうると考えられる読み方が複数生成されていることが表 8 からわかる．このため，人手による評価を行なえば，規則の集合を既存の規則で置き換え新たな規則を生成する手法による性能の向上は，本評価よりも大きくなる可能性がある．

表 8 を見ると，置換することで人に読まれうる，すなわち検索に役に立つと思われる規則が生成されることが分かるが，一方で検索に寄与しない（そう読まれる可能性が少ない）規則も生成されることがわかる．これは検索のスピードとメモリ効率を損なうものの，図 5 の例のような場合の置換ミスをなくす対策を行なえば，検索の性能を損なうものではない．また，確実に使われることはないとは言えないため，検索ミスを減らす意味で全く不要であるとも言えない．

また，表 8 の例よりアルファベットに対して，そのように読まれる可能性が低くとも，あまりにも大きくかけ離れた読みは生成されていないことが推測される．すなわち，規則としてある程度合理的かつ妥当なものが生成されている．このため，これを実際に検索システムに使用した場合，検索精度は下がる可能性があるが，検索精度がほぼ 0 になるような状況にはならないと推定される．

しかし，これが含まれることで別の意図しなかったアルファベット表記とぶつかる可能性が増すことが予測される．衝突自体は正しい規則であっても起こりうる問題であり，これの解消は実際に検索を行なうアプリケーション側で例えば検索の傾向や検索者による判断などで対処する必要がある．

規則を用いる方法のほかに，人名に対する読み全てを網羅的に列挙した辞書として持つ方法もあるが，この辞書を作成することは合理的ではない．また，辞書にない新たな人名に対処することができないという問題もある．

7. 結　　論

本研究では，増田らの分割点を発見することによる対応規則抽出方法により得られた対応規則に対して置換処理を行なうことで，アルファベット表記とカタカナ表記の対データ（人名辞書）と，アルファベット表記の母音を示す文字の知識のみから人手を介さずに自動的に得られる規則の性能の改善を行なった．

置換はカタカナ文字数，アルファベット文字数準拠の 2 通りについて行った．人名辞書データ（対応データ 31847 個）からランダムに 24000 個のテストデータと 7847 個の学習データ（規則抽出対象）を抽出することで作成したデータの組を 10 個作り，実験を行なった．この結果，カタカナ文字数準拠（RRK）では平均値で 0.2856 から 0.2952 に，アルファベット文字数準拠（RRA）では平均値で 0.2856 から 0.3118 に増田らの手法で得られる規則（R）の読みの復元率（元のアルファベット文字列を規則によりカタカナ文字列に完全に復元できたものの内，その読みの集合に正解を含むものの割合）を改良することができた．10 回の内全てにおいて RRA は RRK を上回り，RRK は R を上回った．このため，この結果には統計的有意差があるといえる．

得られた対応規則の数も，10 回の内全てにおいて RRA は最も多く，R は最も少ないという結果を得た．置換により新たに得られた規則の中には人間がそのように読む可能性が十分にある規則が含まれていることを確認した．また，完全に意図されない，すなわち全くランダムに生成されたような読みは見られなかった．

逆読み復元率（元のカタカナ文字列を規則によりアルファベット文字列に完全に復元できたものの内，その読みの集合に正解を含むものの割合）をそれぞれの規則について調べたところ，置換によりこれが低下することが分かった．これについて考察を行い，図 5 のような置換により生成された長いカタカナ文字列が最長一致において優先して適応されるケースにより逆読み復元率が低下することが分かった．また図 7 のようなケースにより逆読み復元率は増加することが分かった．しかしながら増加につながるデータの数よりも低下につながるデータの数が多いため結果として逆綴り復元率は置換により低下した．この図 5 のような置換ミスをなくすことで性能を向上させることができる．これは今後の課題の一つとして挙げられる．

本実験で用いた正解は，表記のゆれを考慮していない人名辞書のデータであるため，人手による評価もしくは別のデータで評価を行なった場合，評価はさらに向上することが，実験の結果より予想される．

本研究は，現在の計算機で実現できる範囲で実験を行い，検討した．今後の課題として，速度やメモリ効率といった性能の向上が挙げられる．

表 8 wadingto に対する読み

R による変換		RRK による変換		
1	2	1		2
ボディント	アン	ボディント	ウォディングト	ン
	イン	ウディグト	ウォディングリット	アン
	オン	ウディグリット	ウォディリングト	イン
	カン	ウディント	ウォディリングリット	オン
	ガン	ウディンリット	ウェイディグト	カン
	クマン	ウディンクト	ウェイディグリット	ガン
	ストン	ウディンクリット	ウェイディント	スン
	スン	ウディングト	ウェイディンリット	セン
	セン	ウディングリット	ウェイディンクト	タン
	タン	ウディリングト	ウェイディンクリット	ダン
	ダン	ウディリングリット	ウェイディングト	トン
	ティン	ワディグト	ウェイディングリット	ドン
	トン	ワディグリット	ウェイディリングト	バン
	ドン	ワディント	ウェイディリングリット	ピン
	バン	ワディンリット		マン
	ピン	ワディンクト		モン
	マン	ワディンクリット		ラン
	モン	ワディングト		リン
	ラン	ワディングリット		ルン
	リアン	ワディリングト		レン
	リン	ワディリングリット		ロン
	ルン	ウォディグト		ンガ
	レン	ウォディグリット		クマン
	ロン	ウォディント		ストン
	ン	ウォディンリット		ティン
	ンガ	ウォディンクト		リアン
	ンセン	ウォディンクリット		ンセン

参 考 文 献

1) 宮内 忠信：カタカナ表記からの英単語検索システムの実現, 情報処理学会研究報告. 自然言語処理研究会報告 93(79), 119-126, 1993-09-16
2) 後藤 功雄 , 加藤 直人 , 田中 英輝 , 江原 暉将 , 浦谷 則好：World Wide Web を用いた外国人名の英訳自動獲得 (自然言語), 情報処理学会論文誌 47(3), 968-979, 2006-03-15
3) 松尾 義博, 白井 諭：発音情報を用いた訳語対の自動抽出, 情報処理学会研究報告. 自然言語処理研究会報告 96(114), 101-106, 1996-11-18
4) 住吉 英樹, 相沢 輝昭：英語固有名詞の片カナ変換, 情報処理学会論文誌 35(1), 35-45, 1994-01-15
5) 飯田 敏幸, 中村 行宏：変形ルールと禁則ルールを用いたカタカナの表記揺らぎの解消法, 情報処理学会論文誌 35(11), 2276-2282, 1994-11-15
6) 伍井 啓恭, 清原 良三, 鈴木 克志, 太細 孝：カタカナ異表記処理, 全国大会講演論文集 第 38 回平成元年前期 (1), 351-352, 1989-03-15
7) 藤井 敦 , 石川 徹也：質問翻訳と文書翻訳を統合した日英言語横断情報検索, 電子情報通信学会論文誌. D-II, 情報・システム, II-パターン処理 J84-D-II(2), 362-369, 2001
8) KNIGHT K. : Machine Transliteration, Computational Linguistics 24(4), 599-612, 1998
9) 増田 恵子, 梅村 恭司：人名辞書から名前読み付与規則を抽出するアルゴリズム, 情報処理学会論文誌 40(7), 2927-2936, 1999-07-15
10) 星野 裕，加藤 博子，永田 健二：８万人西洋人名よみ方綴り方辞典, 日外アソシエーツ（1994）