

Migemo: 日本語のインクリメンタル検索

高林哲^{†‡} 小松弘幸^{*} 増井俊之[†]

[†] ソニーコンピュータサイエンス研究所

[‡] 奈良先端科学技術大学院大学情報科学研究科

^{*} 東京工業大学情報理工学研究科数理・計算科学専攻

[†]{satoru,masui}@csl.sony.co.jp

^{*}komatsu@matsulab.is.titech.ac.jp

インクリメンタル検索は情報検索やテキスト編集などの用途に広く用いられている。しかし、日本語の入力にはかな漢字変換という障壁があるため、キーボードから1文字入力するごとに検索を進めていくスムーズなインクリメンタル検索は従来の手法では行うことができなかった。本稿では、指定された読みで始まる単語をコンパクトな正規表現に動的に展開してインクリメンタル検索を行う手法 Migemo を提案し、これまで困難であった日本語のインクリメンタル検索が実現できることを示す。

Migemo: Incremental Search Method for Japanese Text

Satoru TAKABAYASHI^{†‡} Hiroyuki KOMATSU^{*} Toshiyuki MASUI[†]

[†]Sony Computer Science Laboratories, Inc.

[‡]Nara Institute of Science and Technology, Graduate School of Information Science

^{*}Tokyo Institute of Technology, Graduate School of Information Science
and Engineering, Department of Mathematical and Computing Science

[†]{satoru,masui}@csl.sony.co.jp

^{*}komatsu@matsulab.is.titech.ac.jp

Although incremental search is used in a wide range of tasks including information retrieval and text editing, conventional incremental search method cannot handle Japanese texts effectively because Japanese text entry requires an indirect input method called Kana-Kanji conversion. In this paper, we introduce a new incremental search method called Migemo to realize the incremental search for Japanese texts. Migemo performs the incremental search by dynamically expanding the input pattern into a compact regular expression which represents all the possible words that match the input pattern.

1 はじめに

Emacs などのテキストエディタでは、利用者が 1 文字入力するたびに検索を進めていく、インクリメンタル検索が行える。インクリメンタル検索は、テキスト内から目的の情報を探するという情報検索の用途だけではなく、目的の位置にカーソルをすばやく移動するというテキスト編集の用途にも広く用いられている。図 1 に “scheme” という単語をインクリメンタル検索する過程を示す。

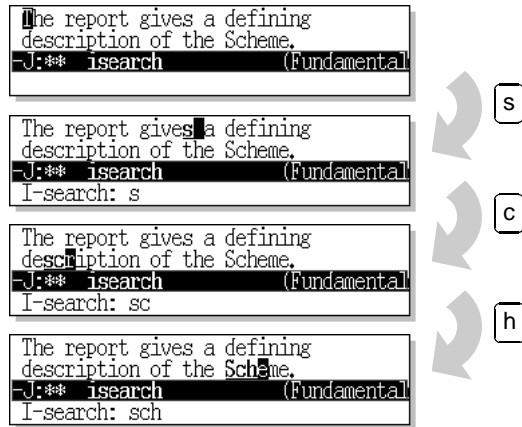


図 1: “scheme” をインクリメンタル検索

この例では、利用者は、`(^S)(s)(c)(h)` の 4 打鍵で目的のテキスト位置に到達している。通常のキーワード検索では最低でも `(s)(c)(h)(e)(m)(e)` の 6 打鍵を要する。インクリメンタル検索を用いれば “floccinaucinihilipilification” のような長い単語でも最初の数文字を打つだけで検索できる。このように、インクリメンタル検索は快適なテキスト編集作業にとって必須の機能といえる。

しかし、日本語でインクリメンタル検索を行おうとすると、かな漢字変換という壁につきあたる¹。インクリメンタル検索では 1 文字入力するごとに検索を進めていくという点が重要だが、かな漢字変換には、ある程度まとまった単位の入力が必要であるため、スムーズな漸進的検索は行えない。一方、英語の場合は、入力する文字とキーボードの対応が一致するため、このような不都合

¹T-Code, TUT-Code といった、漢字を直接に入力する方式もあるが、一般的ではない。

は起きない。一般的に、かな漢字変換には次の 4 つのステップを要する。

1. 読みをローマ字として入力する。
2. ローマ字表記をかな表記に変換する。
3. かな表記を漢字表記に変換する。
4. 変換候補から適切な単語を選択する。

1. ローマ字入力 2. かな表記に変換 3. 漢字表記に変換 4. 変換候補を選択

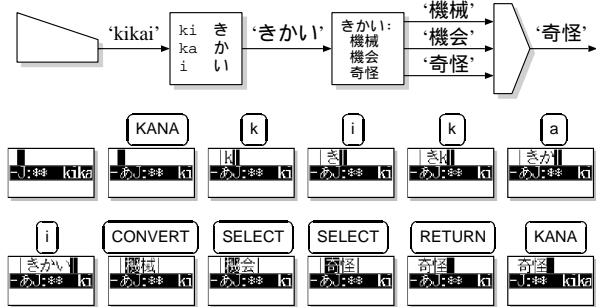


図 2: 「奇怪」を、かな漢字変換で入力

このように、日本語の入力にはかな漢字変換の処理を要するため、従来の手法では「奇怪」をインクリメンタル検索するためには `(^S)(KANA)(k)(i)(k)(a)(i)(CONVERT)(SELECT)(SELECT)(RETURN)(KANA)` のように 10 打鍵以上を要する²。これではインクリメンタル検索の利点は失われている。

2 Migemo

前節で述べた問題を解決するために、我々はインクリメンタル検索の手法 Migemo³ を提案する。Migemo は、かな漢字変換のステップを省略し、ローマ字のままの日本語のインクリメンタル検索を実現する。図 3 に Migemo を用いてキーワード「奇怪」をインクリメンタル検索する過程を示す。

この例では、利用者は `(^S)(k)(i)(k)` という 4 打鍵で目的のテキスト位置に到達している。一方、

²`(^S)`: インクリメンタル検索を開始。`(KANA)`: かな漢字変換を開始。`(CONVERT)`: かな表記を漢字表記に変換。`(SELECT)`: 変換候補から適切な単語を選択。`(RETURN)`: かな漢字変換を終了。

³<http://migemo.namazu.org/>



図 3: 「奇怪」を Migemo でインクリメンタル検索

かな漢字変換を伴う検索では、前述の通り 10 打鍵以上を要する。このように、Migemo では、かな漢字変換の作業に煩わされることなく、スムーズな日本語のインクリメンタル検索が行える。

3 Migemo の実現

Migemo は、利用者が 1 文字入力するごとに動的に正規表現を展開し、指定した読みで始まる複数の言葉を同時かつ並列に検索する。上の「奇怪」のインクリメンタル検索に対する正規表現の展開は図 4 のように行われる。最初の正規表現では“k”にマッチするテキストの位置へ、中央の正規表現では“ki”にマッチする位置へ、最後の正規表現では“kik”にマッチする位置へと、インクリメンタル検索が進む。

このように、実際の検索処理は正規表現エンジンによって行われるが、利用者からは正規表現を展開する過程は隠されているため、通常のインクリメンタル検索とまったく同じように操作が行える。

3.1 正規表現の展開

正規表現の展開は、ヘボン式および訓令式のローマ字列をかな表記に変換する規則と、ローマ字の読みがふられた単語辞書を用いて行われる。

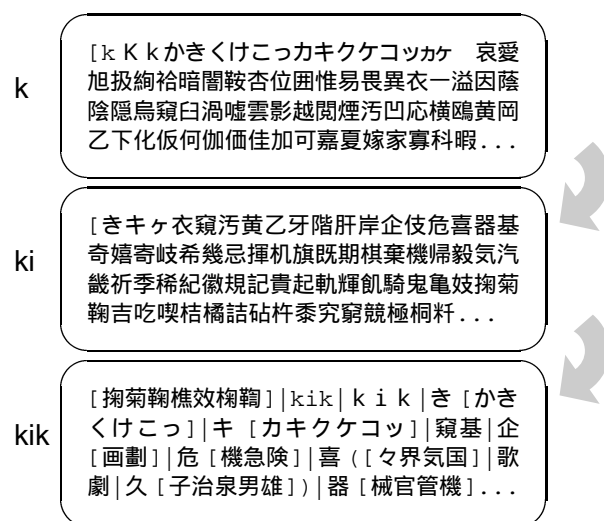


図 4: k, ki, kik に対する正規表現

図 3 の最初の正規表現ではローマ字の変換規則を用いて、“k”に対して次のような展開が行われている。この展開により、“k”でマッチするすべての平仮名および片仮名をカバーできる。

k ⇒ か | き | く | け | こ | っ | カ | キ | ク | ケ | コ | ツ | カケ

単語の展開は図 5 のような辞書を用いて行う。

読み	単語
kikai	機械 機会 奇怪 器械 貴会 気塊 喜界
kikaiabura	機械油
kikaibuhin	機械部品
kikaichinou	機械知能
kikaigakkai	機械学会
kikaigaku	機械学
kikaigakushuu	機械学習

図 5: 正規表現の展開に用いる辞書 (抜粋)

この辞書を用いることにより、指定された読みで始まるすべての単語にマッチする正規表現を動的に生成できる。最も素朴な方法としては、入力 $[k][i][k]$ に対し、“kik”で読みを前方一致検索してマッチした単語をすべて正規表現の or 記号“|”で連結するというものがある。

しかし、マッチする単語を単純に or で連結する手法では、正規表現が巨大になりすぎるといっ

題が起きる。実際、我々の辞書には“k”から始まる単語は同音異義語を含めて47,292個が登録されているため、単純にor記号で結ぶと、実に40万文字もの長さの正規表現が生成されることになり、十分な検索性能は得られない。

そこで、Migemoでは冗長なパターンを統合して正規表現の簡単化を行う。図6に、読み“nez”に対する正規表現の簡単化を示す。

元の正規表現

```
nez|nez|寝|寝|根魚|根崎|寝醒め|根差|根差し|寝惚|寝相|根津|禰津|鼠|鼠|鼠色|鼠男|鼠達|鼠取|捻|捩|螺|捻子|螺子|捩子|ネジ|捻じ伏|捩|捩じ込み|根占|捩じり鉢巻き|捩り鉢巻き|根城|ねざ|ねじ|ねず|ねぜ|ねぞ|ねっ|ネザ|ネズ|ネゼ|ネソ|ネツ
```

簡単化後の正規表現

```
[寝鼠捻螺寝捩捩鼠]|nez|ね [ざじずぜぞっ]|ネ [ザジズゼゾツ]|根 [魚差崎城占津]|禰津|nez
```

図6: “nez”に対する正規表現の簡単化

上の例では、元の正規表現に含まれる「鼠」「鼠色」「鼠男」といった、「鼠」から始まる候補は、長さが最小である「鼠」ひとつにまとめられる。また、「寝」「螺」のような1文字の単語は文字クラス⁴としてまとめられる。さらに「ねざ」「ねじ」「ねず」なども文字クラスを用いて“ね [ざじずぜぞっ]”とまとめられる。このようにして正規表現を簡単化することにより、“k”のような短い読みに対しても数千文字程度の正規表現に収められるため、十分な検索性能が実現できる。

3.2 正規表現展開にかかるコストの軽減

インクリメンタル検索では、利用者にストレスを与えない高速な処理が不可欠である。そこで、Migemoでは、利用者の入力に高速に追従するために、“a”、“k”、“s”などの短い文字列に対しては、

⁴ “[”と“]”で囲まれた [...] という表現

辞書から動的に正規表現を生成する代わりに、あらかじめコンパイルした正規表現を用いて、実用的な速度を実現している。

4 応用

4.1 個人情報管理

我々はPalmPilot用の個人情報管理(PIM)システムQ-Pocket[3]⁵を開発し、その全文検索機能にMigemoを応用している。このシステムでは、テキスト入力システムPOBox[2]用の辞書をそのままMigemoでも利用している。テキスト入力とテキスト検索に同じ辞書を用いることにより、POBoxで入力できる単語はMigemoで必ず検索できるという一貫したテキスト処理が実現できる。たとえば、辞書に“ ”が“star”という読みで登録されていれば、POBoxを用いて[s][t][a][r]というGraffiti操作で“ ”を入力でき、Migemoを用いて[s][t][a][r]というGraffiti操作で“ ”を検索できる。図7に、Q-Pocket上でメモおよびスケジュールデータを検索した結果を示す。文書のタイトルだけでなく、文書の中身を含めて全文検索が行われている。高速な全文検索を実現するために、orで連結された正規表現の検索をAho-Corasick[1]法を用いて最適化している。これはUNIXのfgrepコマンドで採用されている手法である。

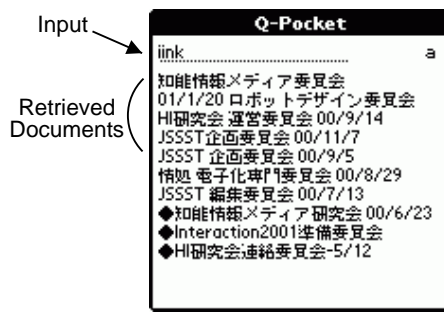


図7: 個人情報のインクリメンタル全文検索

⁵ <http://www.csl.sony.co.jp/person/masui/QPocket/Palm/>

4.2 言語非依存のインクリメンタル検索

これまで日本語のインクリメンタル検索を中心に述べてきたが、Migemo の応用は日本語に限定されるものではない。正規表現の展開に用いる辞書を切り替えることにより、あらゆる言語を扱うことができる。たとえば、

```
jfk John F. Kennedy
```

```
jfk John Fitzgerald Kennedy
```

のような項目を持つ略語辞書を使えば、英文の文書を編集している際に `[j][f][k]` と打鍵して “John F. Kennedy” または “John Fitzgerald Kennedy” をインクリメンタル検索できる。あるいは、

```
header <h1>
```

```
header <h2>
```

```
header <h3>
```

のような項目を持つ辞書を用いれば、HTML 文書を編集している際に、`[h][e][a][d][e][r]` と打鍵して、すべての見出しタグをインクリメンタル検索できる。

また、入力を読みと検索対象が一致しない一風変わった応用も考えられる。図 8 では、

```
hello 你好
```

という項目を含む英中辞書を用いて、“hello” という英単語で中国語の ‘你好’ を言語横断的にインクリメンタル検索している。

```
中文, 普通话, 汉语, 你好
-J:-- migemo (Fundamental)
I-search: hel
```

図 8: 英中辞書を用いた言語横断インクリメンタル検索

4.3 動的単語補完

動的単語補完は、以前に入力したことのある単語を少ない打鍵数で入力するための機構である⁶。たとえば、“floccinaucinihilipilification” を再び入力する際に `[f][l][o][c]` `(EXPAND)` と打鍵すれば “floc” が “floccinaucinihilipilification” に展開される。動

⁶Emacs では dabbrev として実装されている

動的単語補完は打鍵数を減らすだけでなく、打ち間違いを減らすという点でも有効である。

しかし、従来の動的単語補完はインクリメンタル検索と同様に、かな漢字変換という障壁があるため日本語に対しては有効に機能しなかった。そこで、我々は Migemo の正規表現展開の機構を応用してこの問題を解決した。動的単語補完は、カーソル位置からテキストを後ろ向きに検索して補完候補を見つけるといった処理によって行われるが、この検索に Migemo を用いれば、かな漢字変換を省略した動的単語補完が実現できる。図 9 に、Migemo を用いた動的単語補完の例を示す。

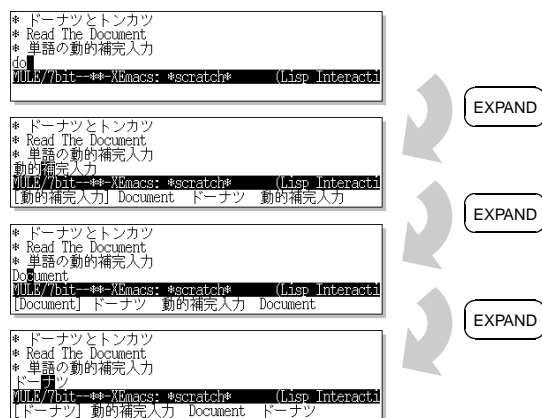


図 9: 動的単語補完: “do” を展開

5 議論

5.1 同音異義語の扱い

Migemo による日本語のインクリメンタル検索では、“kikai” で検索をすると「機械」「機会」「奇怪」「器械」のすべてにマッチするため、「奇怪」を検索しているときに「器械」にマッチしてしまうという問題が起きる。しかし、不適切なマッチは `(S)` で即座にスキップできるため、大きな問題にはならない。一方、通常のキーワード検索では「こんにちば」で検索をかけると「今日は」は見つからないが、Migemo で “konnnitiha” で検索すれば両方とも見つかるという利点がある。同様に、“interface” で検索すれば「インターフェース」「イ

ンターフェイス」「インタフェース」「インタフェイス」などの表記の揺れを無視して検索を行うこともできる。

5.2 辞書への依存

Migemo は辞書を利用して正規表現を生成するため、辞書に載っていない言葉は検索できないという問題がある。また、現在の実装では単語単位の検索にしか対応していないため、“genzainojissou”で「現在の実装」を検索するような、複数の単語にまたがる検索は行えない。この問題は連文節によるかな漢字変換を内部的に行うことで解決できる。通常の日本語入力に用いるかな漢字変換では適切な変換候補を利用者に提示する必要があるが、インクリメンタル検索のための内部的なかな漢字変換では候補の順序を最適化する必要がないため、簡単に実現できると考えている。

5.3 正規表現展開という手法の妥当性

Migemo は入力された読みから検索用の正規表現を展開しているが、日本語のインクリメンタル検索を実現する別の方針として、検索対象のテキストの方をローマ字に変換するという手法が考えられる。

最も素朴なものとして、検索を行うたびにテキストの内容全体を内部的にローマ字列に変換し、そのローマ字列に対して検索を行う方法がある。しかし、この方法はテキストのサイズに比例してローマ字への変換に時間がかかるため、テキストが大きくなると検索性能の点ですぐに破綻する。

次に、検索を行うたびにローマ字へ変換するのではなく、ファイルの読み込みのタイミングなどに一括してローマ字に変換する方法も考えられるが、テキストエディタでは刻々と編集されるテキストと内部的に保持するローマ字列との整合性を取る処理が複雑になる。また、実際のテキストとは別に内部的なローマ字列をメモリ内に保持する必要があるため、メモリ効率の点でも欠点を持つ。

テキスト全体のローマ字列をメモリに保持する

代わりに、カーソル位置から検索を進めつつテキストの必要な部分だけを動的にローマ字列へ変換していく手法も考えられるが、この手法では、後ろ向きにインクリメンタル検索する際に、テキストをローマ字列へ変換する処理が複雑となる。

このように、テキストをローマ字列に変換する手法は実現に多くの困難がある。一方、入力の読みから検索用の正規表現を展開するという我々の手法は簡単に実現できる。

6 結論

我々は、これまで困難であった日本語のインクリメンタル検索を実現する手法 Migemo を提案した。Migemo の応用として PIM システム、言語非依存の検索および動的単語補完の例を紹介した。この他にも、Web ブラウザ w3m への Migemo を実装した日台ら [4] による例がある。将来的にあらゆるソフトウェアに「Migemo 検索」が実現されることを我々は目標としている。

参考文献

- [1] Alfred V. Aho and Margaret J. Corasick. Efficient string matching: an aid to bibliographic search. *Communications of the ACM*, Vol. 18, No. 6, pp. 333–340, 1975.
- [2] Toshiyuki Masui. An efficient text input method for pen-based computers. In *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI '98)*, pp. 328–335. Addison-Wesley, April 1998.
- [3] 増井俊之. 検索と例情報を活用した情報管理手法 Q-Pocket. インタラクティブシステムとソフトウェア VIII: 日本ソフトウェア科学会 WISS2000, December 2000.
- [4] 日台健一, 平岡和幸. w3m 検索機能の強化, 2001. <http://www.me.ics.saitama-u.ac.jp/~hidai/software/w3m/>.