

# ChaSen Technical Report CTR-1

## パトリシア木を用いた形態素解析のための辞書検索 第1版

山下 達雄

奈良先端科学技術大学院大学 情報科学研究科 自然言語処理学講座

tatuo-y@is.aist-nara.ac.jp

1996年10月

### 1 はじめに

この文書では、JUMAN2.0[1] から ChaSen<sup>1</sup> への様々な改良点のうち**形態素辞書検索**について解説する。第2節では日本語形態素解析における辞書引きの手法について説明する。第3節ではJUMAN2.0, 第4節ではChaSenにおける辞書検索アルゴリズムについて説明する。そして、第5節で2つの辞書検索アルゴリズムを比較・検討する。

### 2 日本語形態素の辞書引き

日本語などのわかち書きのされていない言語では単語の境界が自明でないので、英語などのわかち書きされている言語とは辞書を引く対象(文字列)が異なる。

**わかち書きされている言語** 区切られている文字列を単位に辞書を検索する。

(例) This is my pen. → this, is, my, pen を辞書で検索。

**わかち書きされていない言語** 形態素として存在可能な全ての文字列を辞書で検索する。

は し る  
(例) はしる → はし しる を辞書で検索。  
はしる

それゆえ日本語の形態素解析の辞書引きにはトライなどの木構造が用いられる。

### 3 今までの方法 —JUMAN2.0—

JUMAN2.0では、ハッシュデータベース(NDBM)を用いてトライ構造による辞書検索を疑似的に実現している[2]。

例えば「コンビニエンス」という文字列をトライで辞書引きするとき、コ、コン、コンビ、コンビニ、コンビニエ、コンビニエン、コンビニエンスを検索する必要があるが、実際には、コンビ、コンビニ、コンビニエンスしか辞書になく、無駄な検索が多くなってしまいます。

<sup>1</sup>「茶筌」 ChaSen 1.0b1 ( JUMAN 2.0 上位互換 )

そこで、ある長さの文字列で検索したとき次に何文字で検索すれば良いかという情報を付加することにより、無駄な検索を防ぐ(表 1).

辞書検索する文字列	引き出された情報	
	形態素情報(一部)	次に何文字で検索するか?
1. コ	なし	3文字で検索せよ
2. コンビ	普通名詞	4文字で検索せよ
3. コンビニ	普通名詞	7文字で検索せよ
4. コンビニエンス	普通名詞	これでおしまい

表 1: JUMAN2.0 の辞書検索方法

#### 4 パトリシア木を用いた方法 —ChaSen—

パトリシア木とは、ビット単位のトライ(バイナリトライ)中で枝が1本しかない無駄なノードをまとめて圧縮し、効率化を図った木構造である(図 1). 図中のパトリシア木のノード上の数字は、何ビット目をチェックするかを表している. そのビットが1か0かによって辿る枝が異なる. バイナリトライでは全てのビットをチェックするが、パトリシア木ではノード上で指定されたビットしかチェックしないため、最終ノードに到着後にキー全体をマッチングして成否を確認する必要がある. 詳しくは、文献 [3][4][5] を参照されたい.

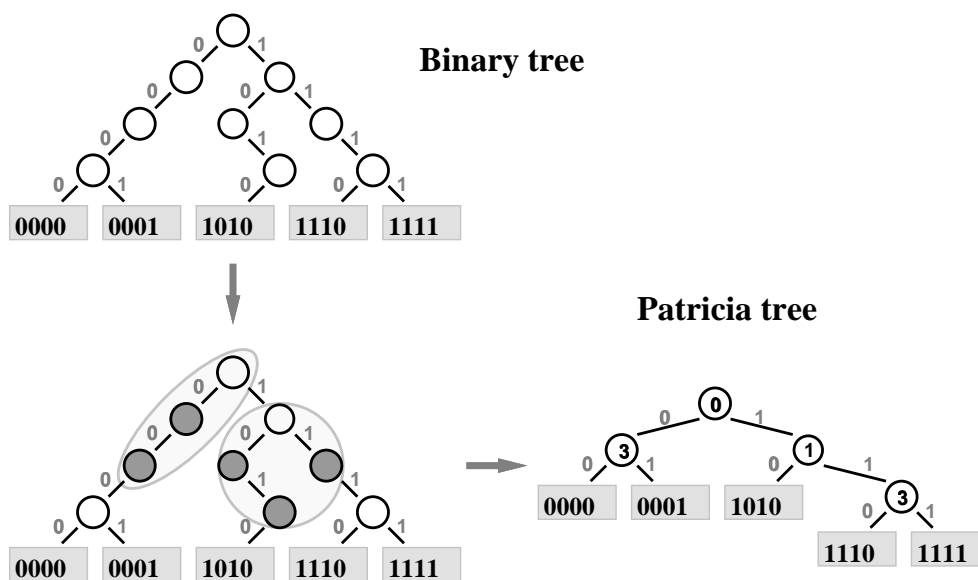


図 1: パトリシア木とは?

さて、パトリシア木を用いると図 2 に示すように、数居ビット(16の倍数)をチェックするノードの左(0)側のノードを見ていくことにより、木を一回探索するだけで、その文字列の先頭から始まる全ての形態素を引き出すことができる(図の例では「あえくれ…」で検索して、「あ」「あえ」「あえくれ」を引き出して

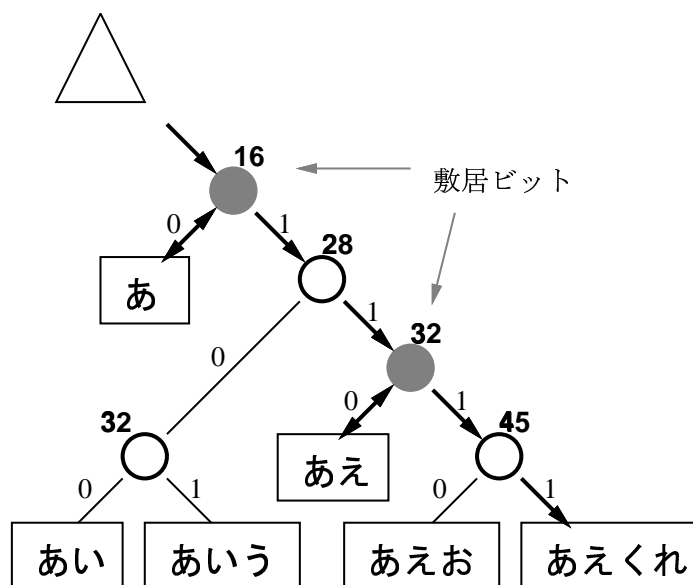


図 2: パトリシア木による辞書引き

いる). 以下に挙げるの 3 つの事実により, 16 の倍数の位置のビットをチェックしているノードがあれば, その長さ (ビット長) の単語がその左 (0) 側に存在することが保証されている.

- 全角文字は 1 文字が 16 ビットで構成される.
- EUC, SJIS では全角文字は 0 ビット目 (一番左) が必ず 1 になる. つまり全角文字列の内部では 16 の倍数の位置のビットは必ず 1 になる.

あい = 10100100101000101010010010100011

- このアルゴリズムでは文字列は後ろに 0 が続く無限ビット列としてパトリシア木に挿入される. つまり全角文字列の外部では 16 の倍数の位置のビットは必ず 0 になる.

あい = 1010010010100010101001001010001100000000...

## 5 比較

JUMAN2.0 の方法 (NDBM) と比べ, パトリシア木を用いると形態素解析時に必要な辞書のサイズが大幅に減る.

また JUMAN2.0 と JUMAN2.0 の辞書検索部分だけをパトリシア木用に変更したもの<sup>2</sup>とで解析速度を比べてみると, 環境によって変動のあるものの確実に速くなっていることが確かめられた. 参考までに表 2 に新聞 3 万文の (当社の環境下での) 解析時間を示す.

パトリシア木検索モジュールの見直しにより更に速くなることが確認されている. この改良は ChaSen に活かされている. 詳しくは文献 [6] を参照のこと.

<sup>2</sup>ChaSen とは異なるプログラムである.

	解析時間 (分)
JUMAN2.0 NDBM ( original )	82
JUMAN2.0 Patricia Tree	58
JUMAN2.0 Patricia Tree & caching	42

表 2: 解析速度の比較

## 参考文献

- [1] 松本裕治・黒橋禎夫・宇津呂武仁・妙木裕・長尾真, 日本語形態素解析システム JUMAN 使用説明書 version 2.0, NAIST Technical Report, NAIST-IS-TR94025, July 1994.
- [2] Sadao Kurohashi, Toshihisa Nakamura, Yuji Matsumoto, Makoto Nagao: Improvements of Japanese Morphological Analyzer JUMAN, Proc. International Workshop on Sharable Natural Language Resources, pp.22-28, August 1994.
- [3] 島内剛一・有澤誠・野下浩平・浜田穂積・伏見正則 編集, “アルゴリズム辞典”, 共立出版株式会社, 1994.
- [4] 菊田昌弘, “用語解説: パトリシアツリー (Patricia Tree),” 人工知能学会誌, Vol.11, No.2, pp.337-339, 1996.
- [5] R. Sedgewick 著 野下浩平・星守・佐藤創・田口東 共訳, アルゴリズム (Algorithms) 原書第2版 第2巻 探索・文字列・計算幾何, 近代科学社, 1992.
- [6] 今一修, “ChaSen 「茶筌」 version 1.0b1 の性能評価 第1版,” ChaSen Technical Report, CTR-2, Oct 1996.