北海道大学
Hokkaido University

情報知識ネットワーク特論
「情報検索とパターン照合」

情報科学研究科 コンピュータサイエンス専攻
情報知識ネットワーク研究室
喜田拓也

第6回
圧縮テキスト上のパターン照合

データ圧縮について
本研究の動機と目標
Huffman符号化テキストに対する照合
LZW圧縮テキストに対する照合
統一的枠組み: Collage system
圧縮による照合の高速化という視点: BPE圧縮

データ圧縮について

可逆圧縮
（lossless compression）

非ユニバーサル符号
- Run-length

エントロピー符号
- ハフマン符号
- 算術符号化

ユニバーサル符号
- 辞書式
  - LZ78
  - LZ77
  - LZW
- 文法圧縮
  - Sequitur
  - BPE
- 統計型
  - PPM
- sort-based
  - BWT

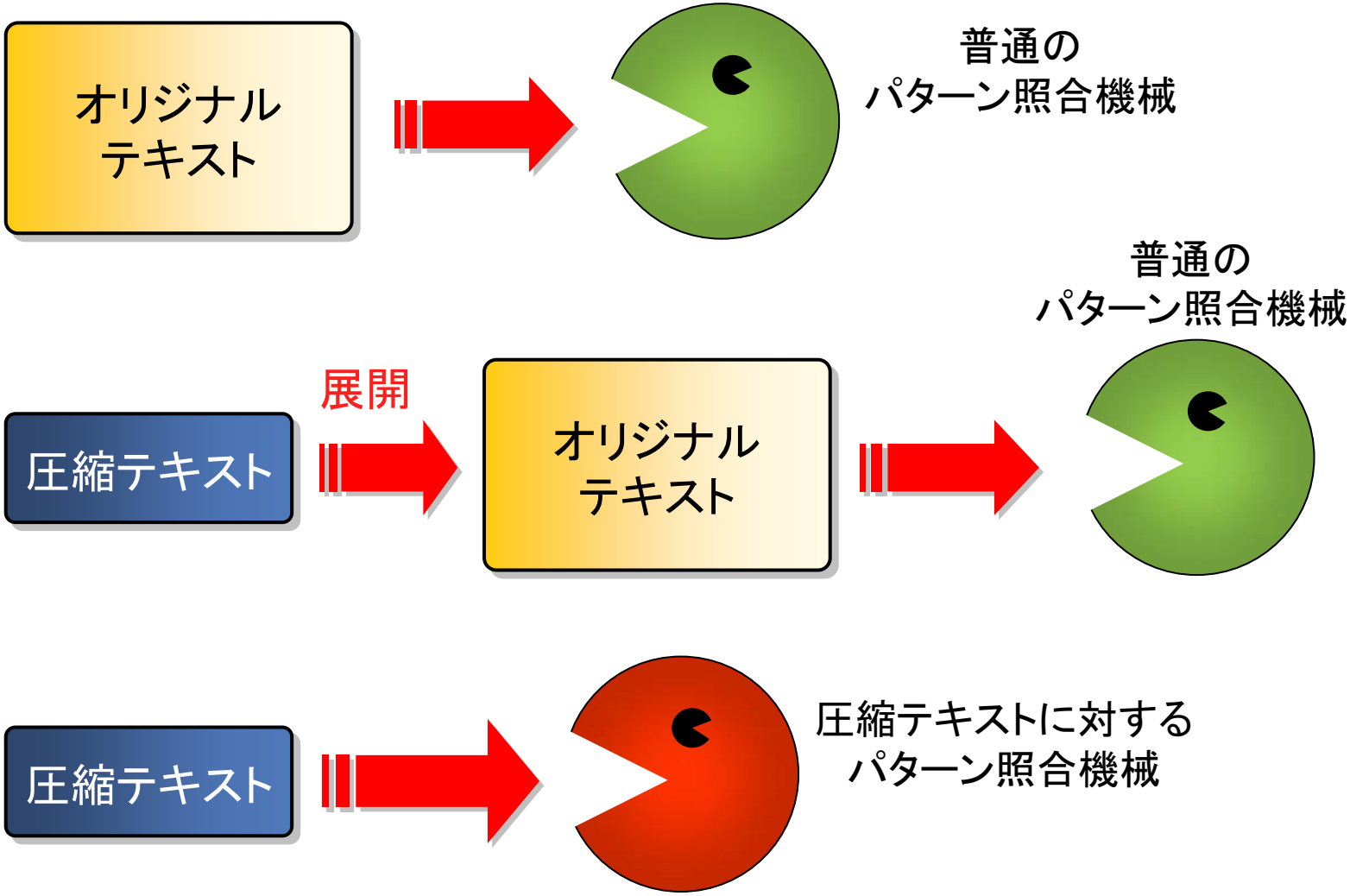
非可逆圧縮
（lossy compression）

- JPEG
- MPEG
- MP3

映像や音声のデータに用いられる

※参考文献：Managing Gigabytes: Compressing and Indexing Documents and Images, I. H. Witten, A. Moffat, T. C. Bell, Morgan Kaufmann Pub, 1999.

本研究の目的

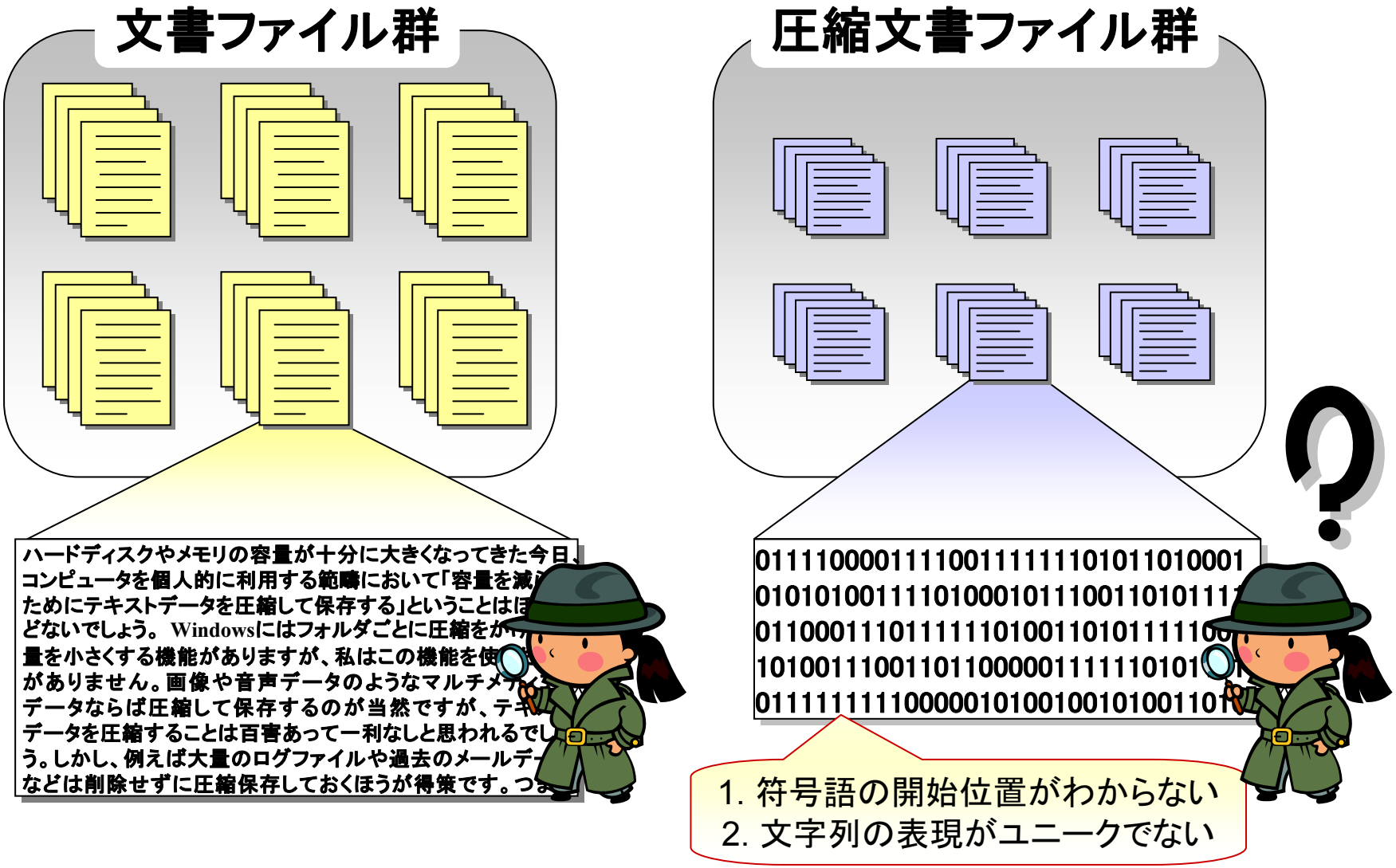

オリジナル
テキスト
普通の
パターン照合機械
普通の
パターン照合機械
展開
圧縮テキスト
オリジナル
テキスト
圧縮テキスト
圧縮テキストに対する
パターン照合機械

本研究の応用例

- 小型のコンピュータ(携帯電話やPDA等)に
  できるだけ多くのデータを詰め込みたい！
- メモリが小さいので、索引データ構造は作りたくない！
- しかし、検索は高速に行いたい！

※写真はsharp mi110と東芝V601T

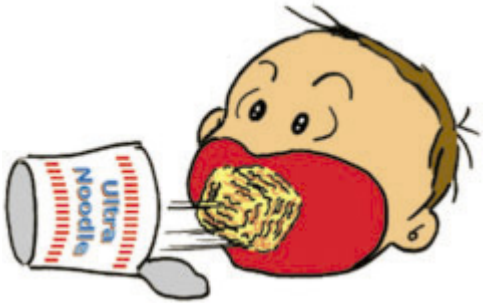
圧縮されたテキストに対する照合の難しさ



本研究の目標

「展開してから」法

「展開しながら」法



「展開しないで」法

目標： これらより速い！

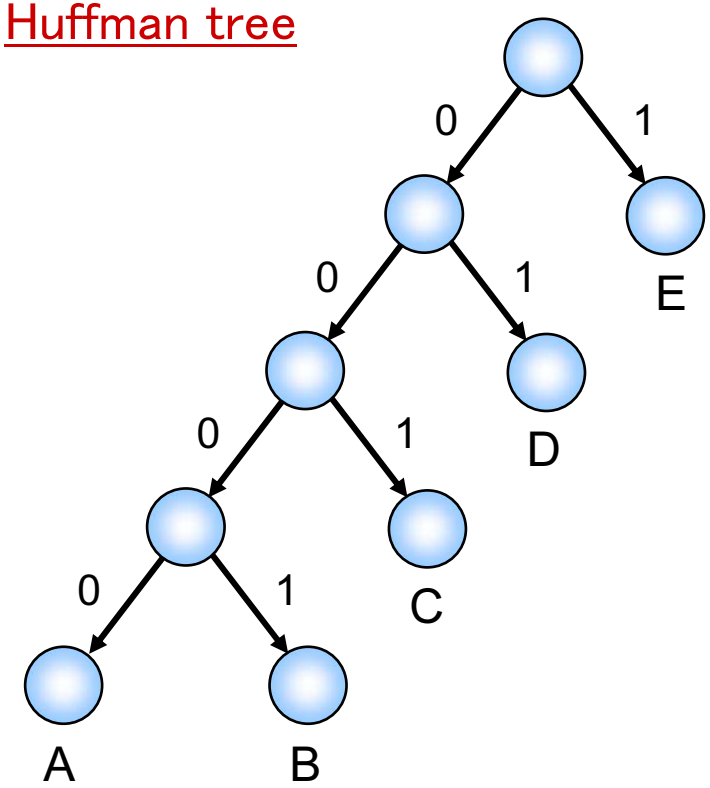
※ 上図イラストは竹田正幸先生の作；出版物初出「情報処理」2002年, Vol.43, No.7, p.764(図-1)

本分野における研究

圧縮方法	照合アルゴリズム
Run-length	Eilam-Tzoreff & Vishkin (1988)
Run-length (two dim.)	Amir *et al.* (1992, 1997); Amir & Benson (1992)
LZ77 family	Farach & Thorup (1995); Gąsieniec, *et al.* (1996); Klein & Shapira (2000)
LZ78 family	Amir *et al.* (1996); Kida *et al.* (1998, 1999); Navarro & Tarhio (2000); Kärkkäinen *et al.* (2000);
LZ family	Navarro *et al.* (1999)
Straight-line programs	Karpinski *et al.* (1997); Miyazaki *et al.* (1997); Hirao *et al.* (2000)
Huffman	Fukamachi *et al.* (1998); Klein & Shapira (2001); Miyazaki *et al.* (1998)
Finite state encoding	Takeda (1997)
Word based encoding	Moura *et al.* (1998)
Pattern substitution	Manber (1994); Shibata *et al.* (1998)
Antidictionary based	Shibata *et al.* (1999)

Huffman符号化テキスト上の照合

Huffman tree



パターン P = DEC

Huffman符号化した
パターン E(P) = 011001

テキスト T = ABECA・・・

Huffman符号化した
テキスト E(T) = 0000000110010000・・・

誤った出現！

同期を取る方法

- 素朴な方法
  - ビットを読むごとにHuffman treeを(根から順に)たどりながら符号語の先頭位置を判別し、任意のPrefix型アルゴリズムでパターン照合を行う
  - 符号語の先頭かどうかを判定する手間がかかるので、遅い
- フィルタリング＋検査
  - Shmuel T. Klein, D. Shapira:
    Pattern Matching in Huffman Encoded Texts. DCC2001: 449-458.
  - E(T)からE(P)を任意のパターン照合アルゴリズムを用いて検索し、ヒットした位置が正しい出現かどうかを調べる
  - Huffman符号語の場合、符号語の途中から復号しても、ある程度のビットを読めば自動的に同期が回復する。よって、ヒットした位置から定数Kビットだけ手前に戻ってからヒットの語頭位置を検査する
    - Kはだいたい100もとれば、十分に語検出は抑えられる
    - パターンが長くなればなるほど語検出しにくくなる

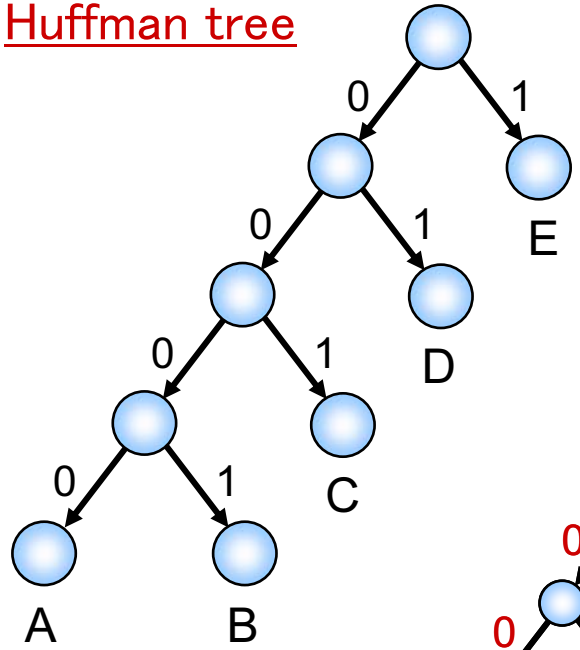
実験的に実証

- 同期型KMP(AC)オートマトン
  - M. Miyazaki, S. Fukamachi, M. Takeda:
    Speeding up the pattern matching machine for compressed texts (in Japanese), Trans. IPSJ, Vol. 39, No. 9, pp.2638-2648, 1998.

同期つきオートマトンによる解決

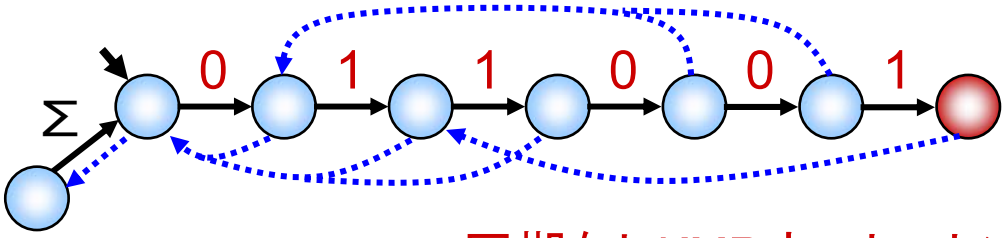
M. Miyaaki, S. Fukamachi, M. Takeda: Speeding up the pattern matching machine for compressed texts (in Japanese), Trans. IPSJ, Vol. 39, No. 9, pp.2638-2648, 1998.

Huffman tree

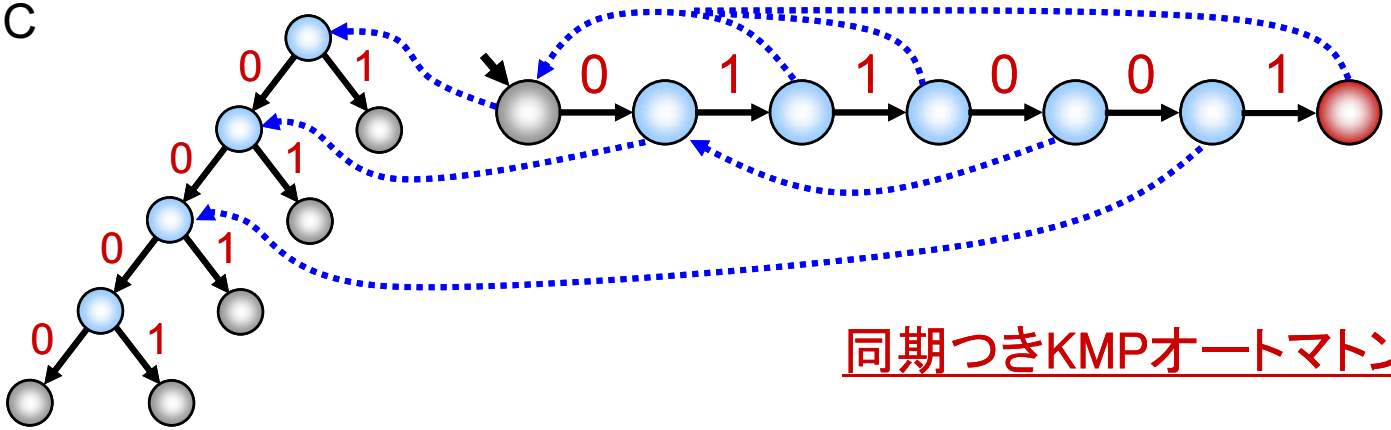


パターン P = DEC

Huffman符号化した
パターン E(P) = 011001



同期なしKMPオートマトン



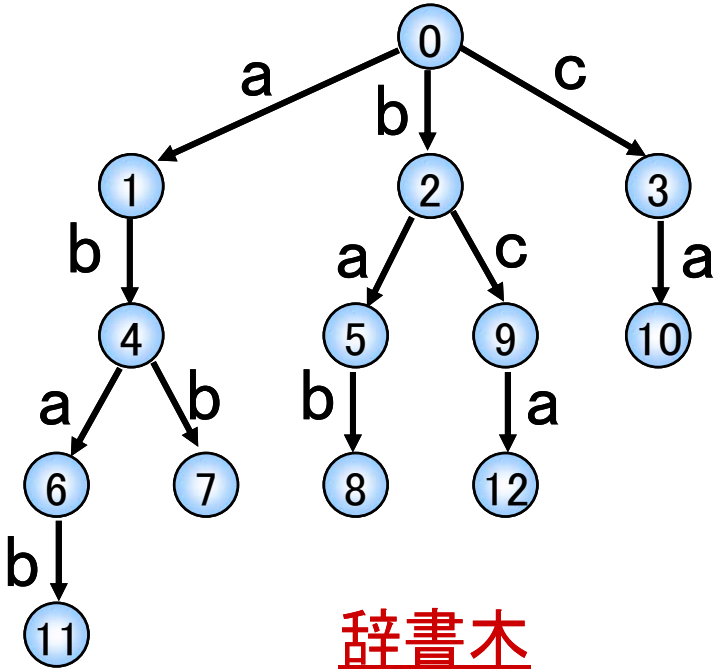
同期つきKMPオートマトン

テキストT = ABECA・・・　Huffman符号化テキストE(T) = 0000000110010000・・・

Lempel-Ziv-Welch (LZW)圧縮法

T. A. Welch: A technique for high performance data compression, IEEE Comput., Vol.17, pp.8-19, 1984.

テキスト T:	a	b	ab	ab	ba	b	c	aba	bc	abab
圧縮テキスト E(T):	1	2	4	4	5	2	3	6	9	11



辞書木

辞書木に登録されている文字列の集合をDとする

D = {a, b, c, ab, ba, bc, ca, aba, abb, bab, bca, abab}

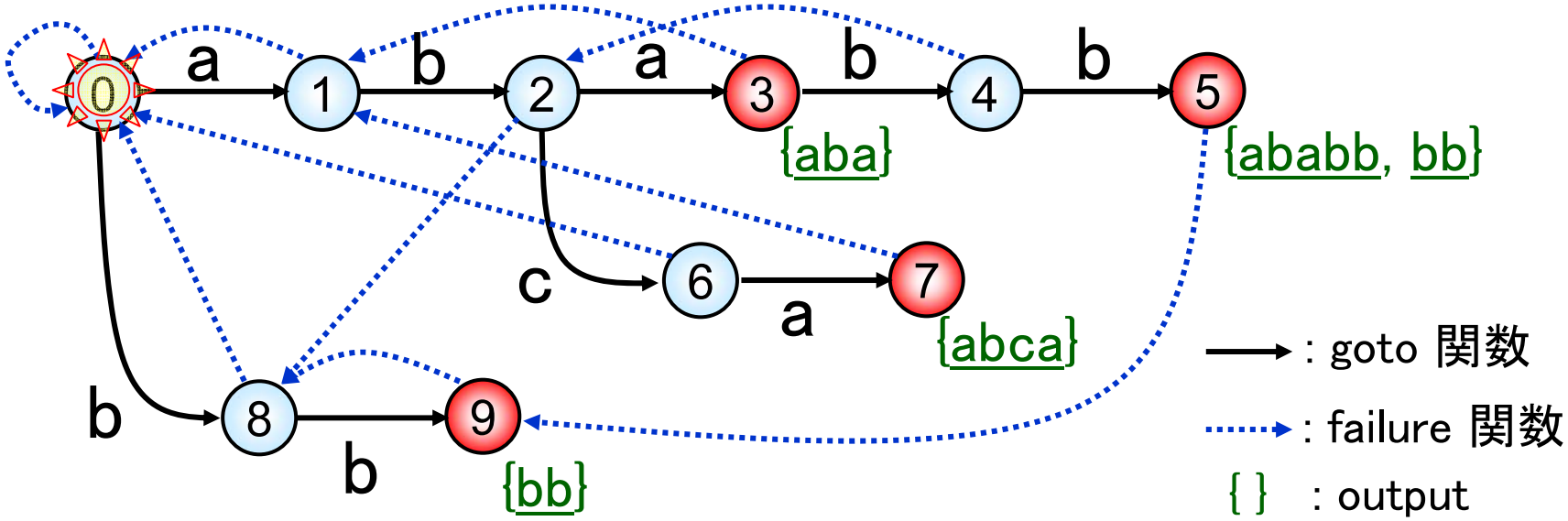
Dは適応的に構築される

|D| = O(圧縮テキスト長)

※ UNIXのcompressコマンドやGIF形式の画像圧縮などに使われている

復習：Aho-Corasick照合機械の動作

パターン集合 Π = {aba, ababb, abca, bb} に対するAC照合機械



状態： 0 → 1 → 2 → 3 → 4 → 3 → 4 → 5 → 1

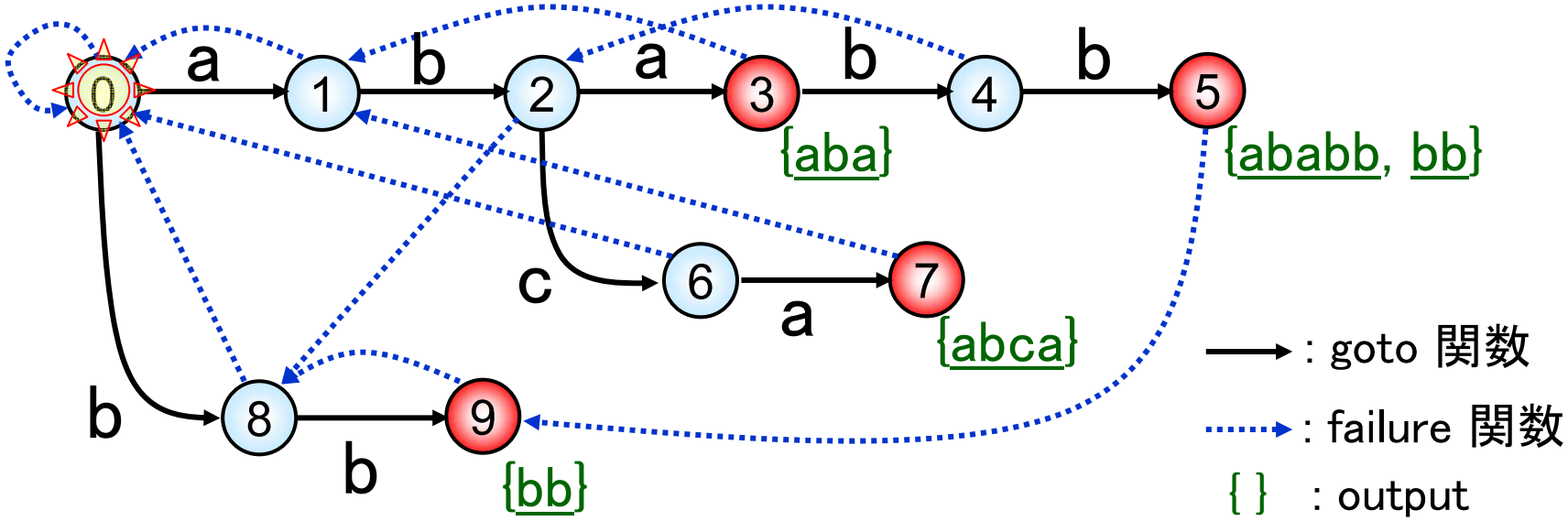
テキスト： a b a b a b b a

Output： aba aba ababb bb

アルゴリズムのアイデア

T. Kida, M. Takeda, A. Shinohara, M. Miyazaki, and S. Arikawa: Multiple pattern matching in LZW compressed text, Proc. Data Compression Conference, pp. 103-112, IEEE Computer Society, Mar. 1998.

- LZW圧縮テキスト上で、AC照合機械の動作をシミュレートしたい



状態:	0	→	1	→	2	→	3	→	4	→	3	→	4	→	5	→	1
テキスト:			a		b		a		b		a		b		b		a
圧縮テキスト:			1		2		4				4				5		
Output:							aba				aba				bb ababb		

アルゴリズムの核となる関数：Jump & Output

- 二つの関数 Jump と Output をうまく計算できないか？
  - 関数 Jump(q, u)：
    - 文字列 u に対応する遷移の連続をO(1)時間で模倣する
    - 定義域はQ×D
    - AC照合機械の状態を返す

単純には O(m|D|)領域が必要と考えられる

O(m^2+|D|)領域で実現！

  - 関数 Output(q, u)：
    - 状態 q に対応する文字列と u を連結した文字列中に現れるパターンの出現位置をO(r)時間で報告する
    - 定義域はQ×D
    - パターンの集合を返す

単純には O(m|D|)領域が必要と考えられる

O(m^2+|D|)領域で実現！

関数Jumpについて

δ(q,u) をもとのAC照合機械の(拡張された)状態遷移関数※とすると

O(m^3)領域

$$\mathrm{Jump}(q, u) = \begin{cases} \delta(q, u) & u\text{があるパターンの部分文字列のとき} \\ \delta(\varepsilon, u) & \text{それ以外のとき} \end{cases}$$

O(|D|)領域

O(m^2)領域　O(m^2)領域※

$$\mathrm{Jump}(q, u) = \begin{cases} \mathrm{Ancestor}(N'_1(q, u'), |u'| - |u|) & u\text{があるパターンの部分文字列のとき} \\ \delta(\varepsilon, u) & \text{それ以外のとき} \end{cases}$$

O(|D|)領域

※ つまりδ(q,u)は、状態qから文字列uを読み込んだとき、gotoとfailureで遷移した先の状態を返す関数
※ u' はPに対するgeneralized suffix trie上で、uの先祖で最も近いexplicitなノードに対応する文字列

関数Outputについて

$\tilde{u}$:uの接頭辞でかつパターンの接尾辞である最長の文字列

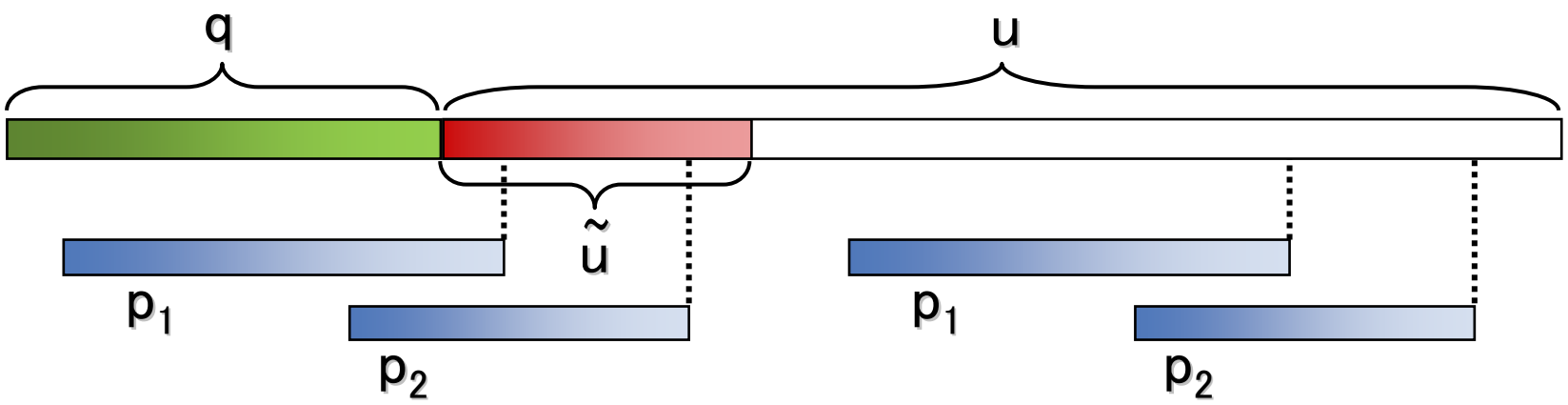
$$A(u) = \{\langle i,p\rangle \mid p \in \Pi,\ |\tilde{u}| < i < |u|,\ |p| < i,\ \text{and } u[i-|p|+1 \ldots i] = p\}$$

状態qは、あるパターンのprefixに対応することに注意

$O(m^2)$領域

$O(|D|)$領域

$$\mathrm{Output}(q, u) = \mathrm{Output}(q, \tilde{u}) \cup A(u)$$



Kida, et al. [1998]の結果

- おおもとのアイデア：
  - A. Amir, G. Benson, and M. Farach: Let sleeping files lie: Pattern matching in Z-compressed files, J. Computer and System Sciences, Vol.52, pp.299-307, 1996.
    - LZW圧縮テキスト上でKMPをシミュレート
- Aho-Corasick（AC）照合機械を模倣することにより、複数パターンを同時に探索することが可能
- 前処理に$O(m^2 + |\Sigma|)$時間・領域かかる
- 任意の個数のパターンについて、$O(n+r)$時間・ $O(m^2+|D|)$領域で圧縮テキストを走査し、すべての出現位置を報告する

※ 初出は、「T. Kida, M. Takeda, A. Shinohara, M. Miyazaki, and S. Arikawa: Multiple pattern matching in LZW compressed text, Proc. Data Compression Conference, pp. 103-112, IEEE Computer Society, Mar. 1998. 」だが、Journal版が2000年にある：「T. Kida, M. Takeda, A. Shinohara, M. Miyazaki, and S. Arikawa: Multiple Pattern Matching in LZW Compressed Text, Journal of Discrete Algorithms, 1(1), pp. 133-158, Hermes Science Publishing, Dec. 2000.」

Bit-parallel化のアイデア

T. Kida, M. Takeda, A. Shinohara, and S. Arikawa: Shift-And approach to pattern matching in LZW compressed text, Proc. CPM'99, LNCS1645, pp. 1-13, Springer-Verlag, Jul. 1999.

パターンP:= a a b a c

テキストT:= a a b a a c a a b a c a b

	a	a	b	a	a	c	a	a	b	a	c	a	b
a	1	1	0	1	1	0	1	1	0	1	0	1	0
a	0	1	0	0	1	0	0	1	0	0	0	0	0
b	0	0	1	0	0	0	0	0	1	0	0	0	0
a	0	0	0	1	0	0	0	0	0	1	0	0	0
c	0	0	0	0	0	0	0	0	0	0	1	0	0

マスクビット

a	b	c
1	0	0
1	0	0
0	1	0
1	0	0
0	0	1

Jump! *Jump!*

拡張した状態更新関数f'

- 任意の $a \in \Sigma$, $u \in \Sigma^*$, $S \in \{1, \cdots, m\}$ について以下のように定義する
  - $M(a) = \{ 1 < i < m \mid P[i] = a \}$
  - $f(S, a) = ((S \oplus 1) \cup \{1\}) \cap M(a)$
  - $f'(S, \varepsilon) = S$ and $f'(S, ua) = f'(f(S, u), a)$
  - $M'(u) = f'(\{1, \cdots, m\}, u)$

O(|D|)時間・領域

- このとき、任意の$u \in \Sigma^*$, $S \in \{1, \cdots, m\}$ について
  - $f'(S, u) = ((S \oplus |u|) \cup \{1, \cdots, |u|\}) \cap M'(u)$

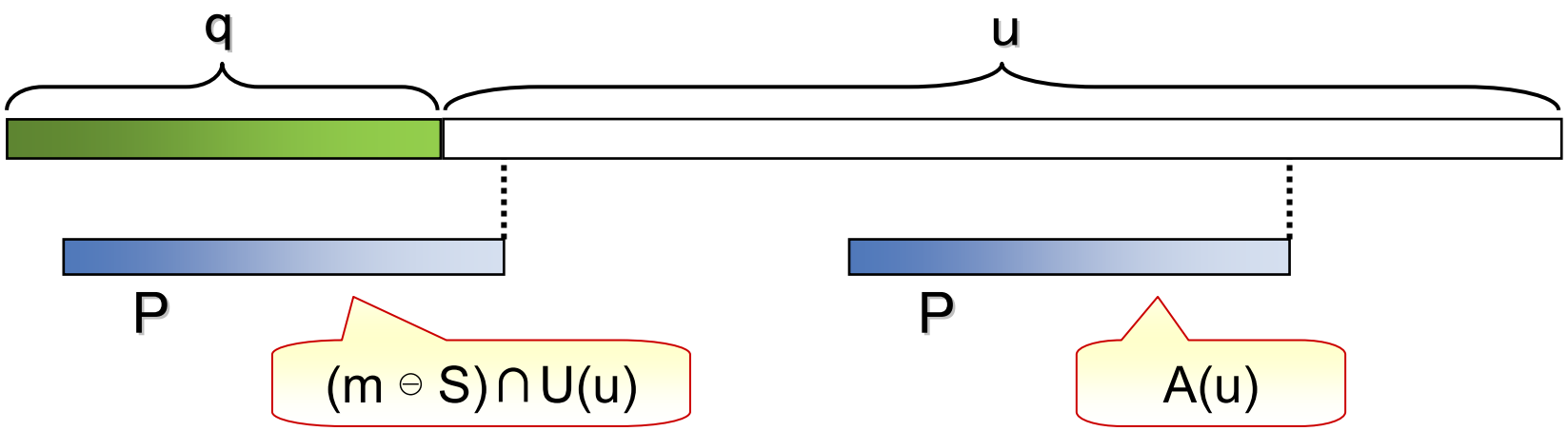
O(1)時間

関数OutputのBit-parallel版

- 定義:
  - Output(S, u) = { 1 < j < |u| | m∈S }
  - U(u) = {1 < j < |u| | i <m かつ u[1..i] =P[m−i+1..m] }
  - A(u) = {1 < j < |u| | m < i かつ u[1−m+1..i]=P }
  - Output(S, u) =((m ⊖ S)∩U(u)) ∪ A(u)

O(|D|)時間・領域

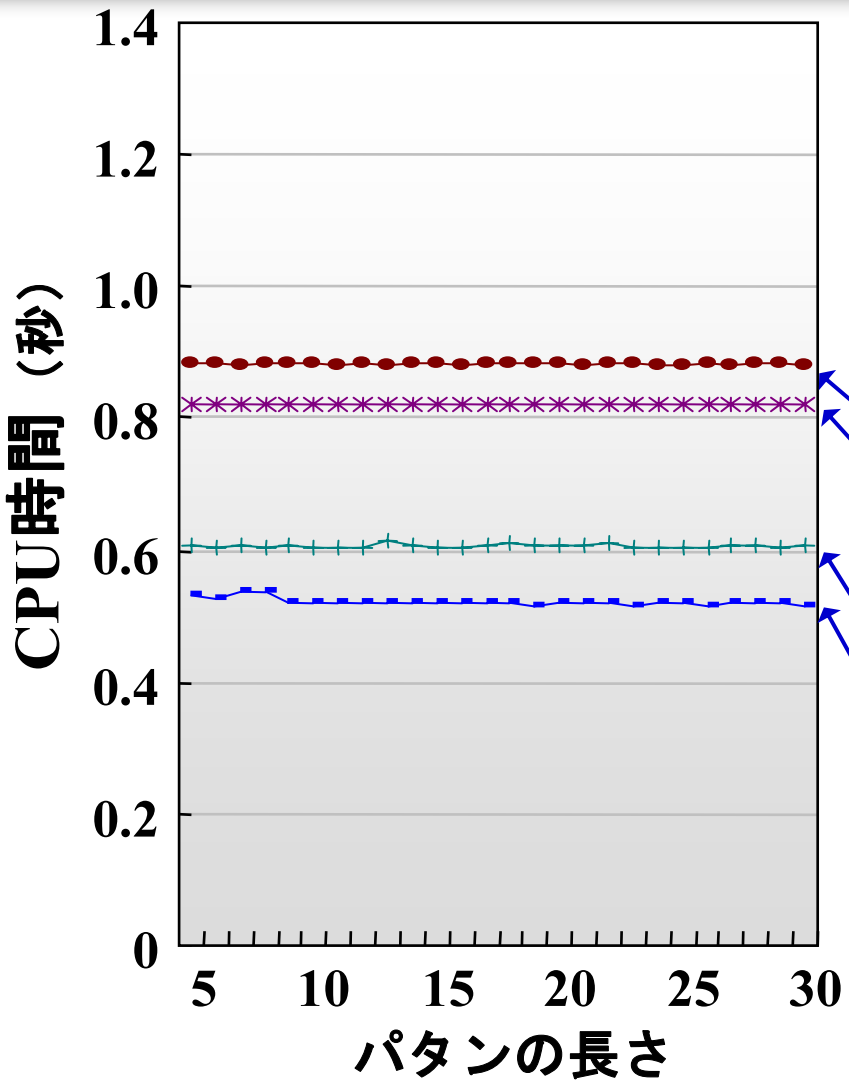
O(|D|)時間・領域



Kida, et al. [1999]の結果

- Shift-And法に基づいたBit-parallel手法を適用して、状態遷移関数とOutput関数の処理を高速化
- 前処理にO(m+|Σ|)時間・領域を使用
- 一つのパターンについて、O(n+r)時間・O(m+|D|)領域で圧縮テキストを走査し、すべての出現位置を報告する
- Shift-And法と同様、拡張性に優れている
  - 一般化されたパターンに対する文字列照合
  - k個のミスマッチを許した文字列照合
  - 複数パターンへの対応

目標の達成！



AlphaStation XP1000
(Alpha21264: 667MHz)
Tru64 UNIX V4.0F

Genbank（DNA塩基配列）
17.1Mbyte

「展開しながら」法

- compress(LZW)+**KMP**
- gunzip(LZ77)+**KMP**

「展開しないで」法

- T. Kidaら[1998]
- ビットパラレルによる高速化[1999]

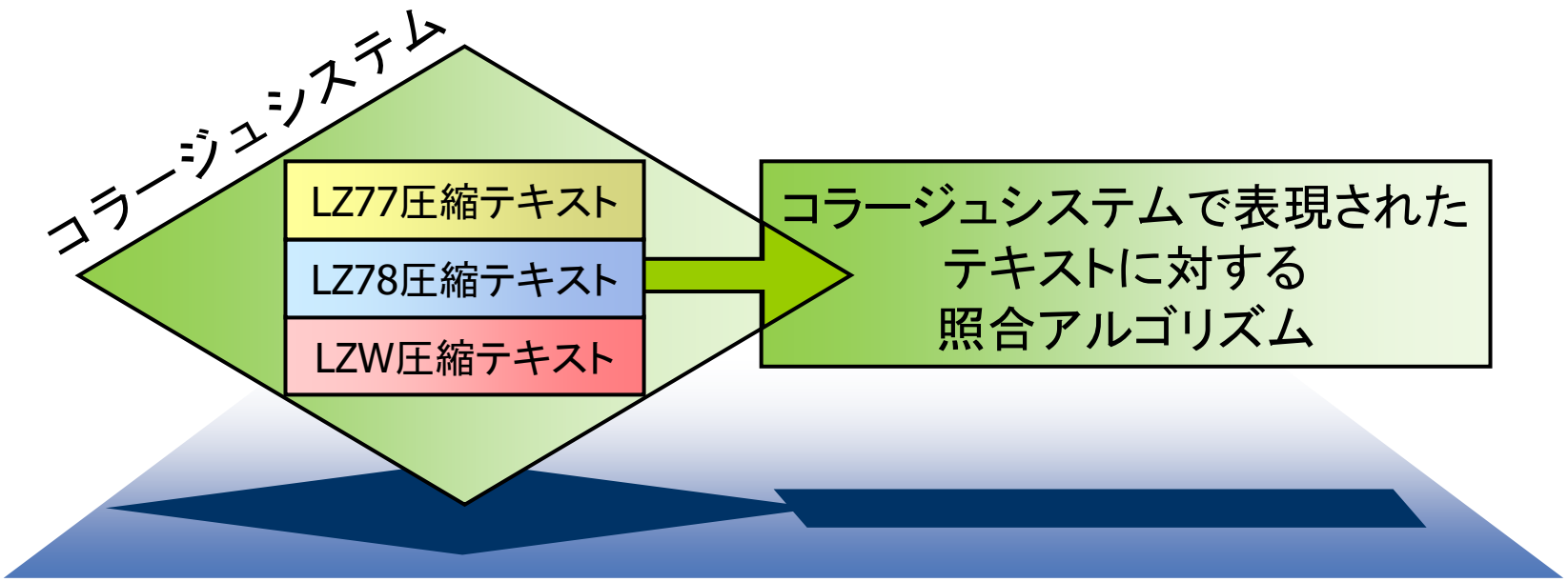
論文の衝突！

- 第一次ショック！（at CPM’99）
  - T. Kida, et al., Shift-And Approach to Pattern Matching in LZW Compressed Text
  - G. Navarro and M. Raffinot, A General Practical Approach to Pattern Matching over Ziv-Liempel Compressed Text
- 第二次ショック！（at CPM2000）
  - Y. Shibata, et al., A Boyer-Moore type algorithm for compressed pattern matching
  - G. Navarro and J. Tarhio, Boyer-Moore string matching over Ziv-Lempel compressed text

コラージュ・システム(Collage system)

T. Kida, et al. A Unifying Framework for Compressed Pattern Matching, In Proc. 6th International Symp. On String Processing and Information Retrieval, pp. 89-96. IEEE Computer Society, 1999.

- 辞書式圧縮法によって圧縮されたテキストを統一的に表現するための形式的体系



コラージュ・システムの例

$\mathcal{D}$:

$X_1 = a$;

$X_2 = b$;

$X_3 = X_1 \cdot X_2$; → (連接) a b

$X_4 = X_2 \cdot X_1$; → (連接) b a

$X_5 = (X_3)^3$; → (繰り返し) a b a b a b

$X_6 = {}^{[3]}X_5$; → (切り落とし) b a b

$\|\mathcal{D}\| = 6$

$\mathcal{S}$: $X_3\, X_6\, X_4\, X_5\, X_2\, X_3\, X_1\, X_5\, X_4\, X_2$

$|\mathcal{S}| = 10$

a b | b a b | b a | a b a b a b | b | a b | a | a b a b a b | b a | b

コラージュ・システムの定義

- コラージュシステムとは，組〈D, S 〉
- D： トークン割当の集合（辞書に相当）
  - $X_1 = expr_1$; $X_2 = expr_2$; ・・・ ; $X_n = expr_n$
    各 $expr_k$ は以下のいずれか
    - a　　$a \in \Sigma \cup \{\varepsilon\}$　　一文字割当
    - $X_i \cdot X_j$　　i, j は整数で，i, j ＜ k　　連結
    - $(X_i)^j$　　i, j は整数で，i ＜ k　　繰り返し
    - ${}^{[j]}X_i$　　i, j は整数で，i ＜ k　　前切り落とし
    - $X_i^{[j]}$　　i, j は整数で，i ＜ k　　後切り落とし
  - ||D|| = n ： D中のトークンの個数
  - X.u ：トークン X が表す文字列
- S： D で割当てられたトークンの列（符号列に相当）
  - $X_{i1} X_{i2} \cdots X_{il}$　　（X_i は D中のトークン）
  - |S| = l ： トークンの列の長さ

照合アルゴリズムの計算量

定理

コラージュシステム$\langle \mathcal{D}, \mathcal{S} \rangle$で表現されたテキストに対する文字列照合問題は,
$O(\|\mathcal{D}\| + |\mathbf{P}|^2)$ 領域を用いて
$O((\|\mathcal{D}\| + |\mathcal{S}|) \cdot height(\mathcal{D}) + |\mathbf{P}|^2 + r)$ 時間で解決できる.

圧縮テキスト長

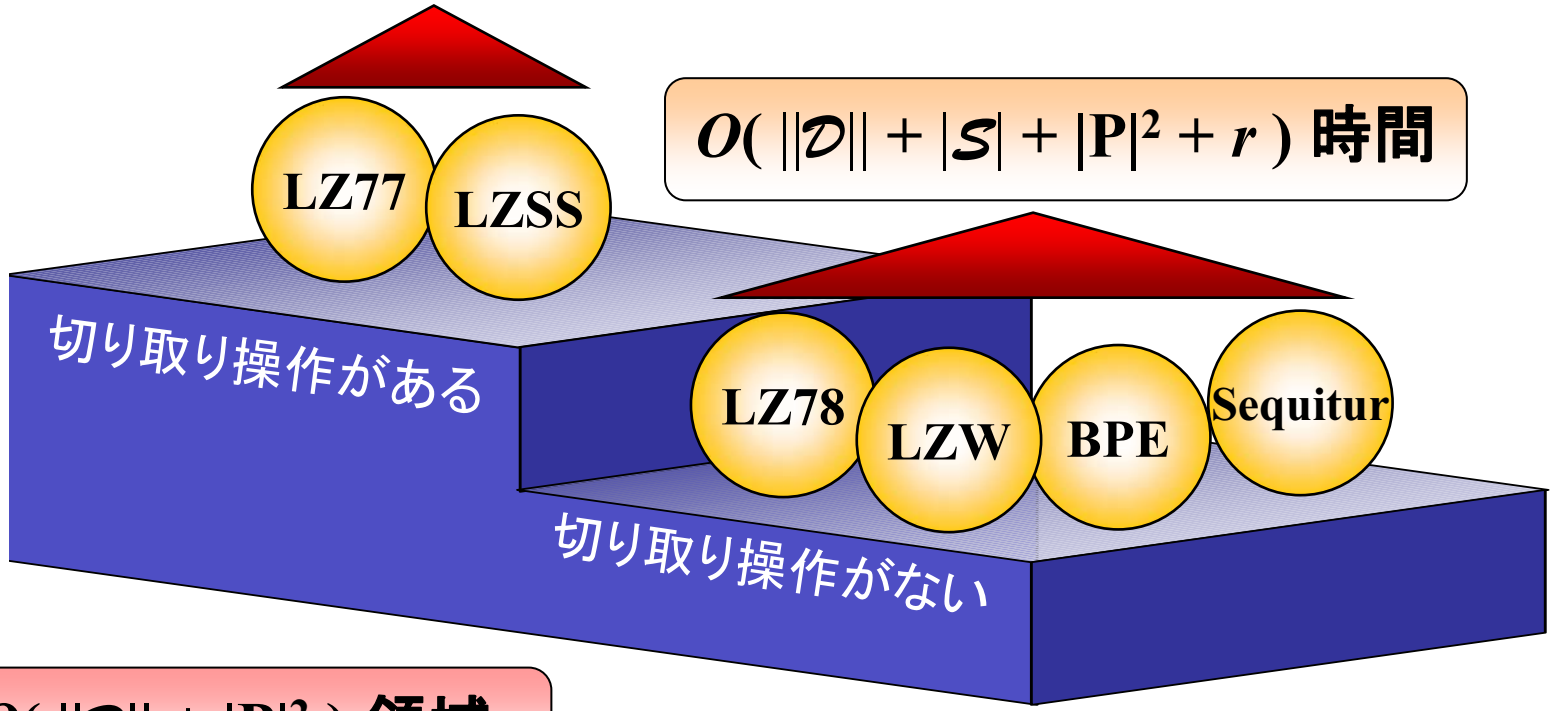
$|\mathbf{P}|$ はパタンの長さ

r はパタンの出現回数

$\mathcal{D}$が切り落とし操作を含まない場合は,
$O(\|\mathcal{D}\| + |\mathcal{S}| + |\mathbf{P}|^2 + r)$ 時間で解決できる.

得られた知見

$O(\,(\|\mathcal{D}\|+|\mathcal{S}|)\cdot height(\mathcal{D}) + |P|^2 + r\,)$ 時間



$O(\,\|\mathcal{D}\| + |\mathcal{S}| + |P|^2 + r\,)$ 時間

$O(\,\|\mathcal{D}\| + |P|^2\,)$ 領域

ちょっと、ひといき・・・

- ここまでのまとめ
  - Huffman符号化テキストに対する照合 → 同期つきオートマトン
  - LZW圧縮テキストに対する照合 → KMP(AC)の遷移をLZW上で模倣
  - 統一的枠組み: Collage system
    - 辞書式圧縮法によって圧縮されたテキストを統一的に表現するための形式的体系
    - Collage systemの枠組みに入る圧縮法に関して、その上での統一的なパターン照合アルゴリズムを得た
    - 切り取り操作を含まない圧縮法は、パターン照合に向いている

キングペンギンの雛（旭山動物園にて '05.3.13）

～トリビア～

佐藤雅彦 著:「毎月新聞」より

左を見よ

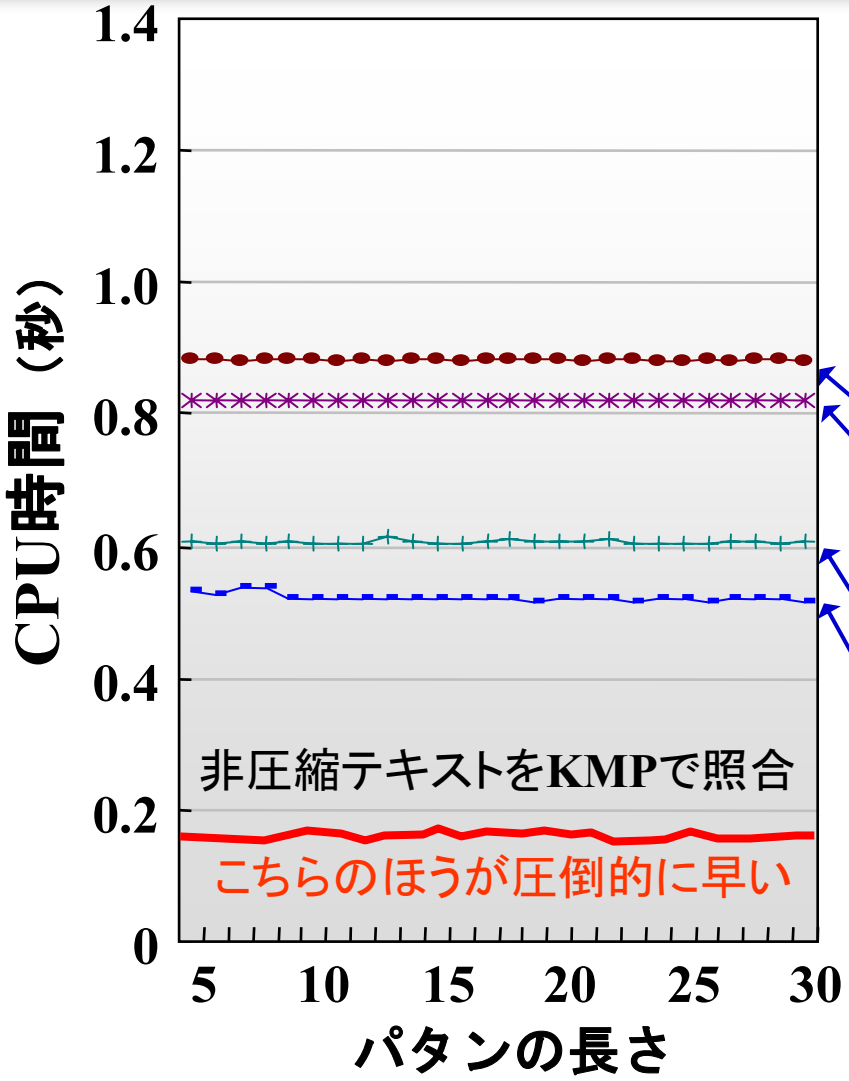
圧縮文字列照合する理由は？

容量は十分あるのに、
テキストを圧縮して
保存しますか？

もし・・・ **目標2** 新たな目標

元のテキスト上での照合時間 > **圧縮テキスト上の照合時間**

新たな目標（目標2）



AlphaStation XP1000
(Alpha21264: 667MHz)
Tru64 UNIX V4.0F

Genbank（DNA塩基配列）
17.1Mbyte

「展開しながら」法

- compress(LZW)+**KMP**
- gunzip(LZ77)+**KMP**

「展開しないで」法

- T. Kidaら[1998]
- ビットパラレルによる高速化[1999]

Bite Pair Encoding(BPE)圧縮法

18

テキスト ABABCDEBDEFABDEABC

↓ G

GGCDEBDEFGDEGC

↓ H

GGCHBHFGHGC

↓ I

置換後の文字列 GIHBHFGHI

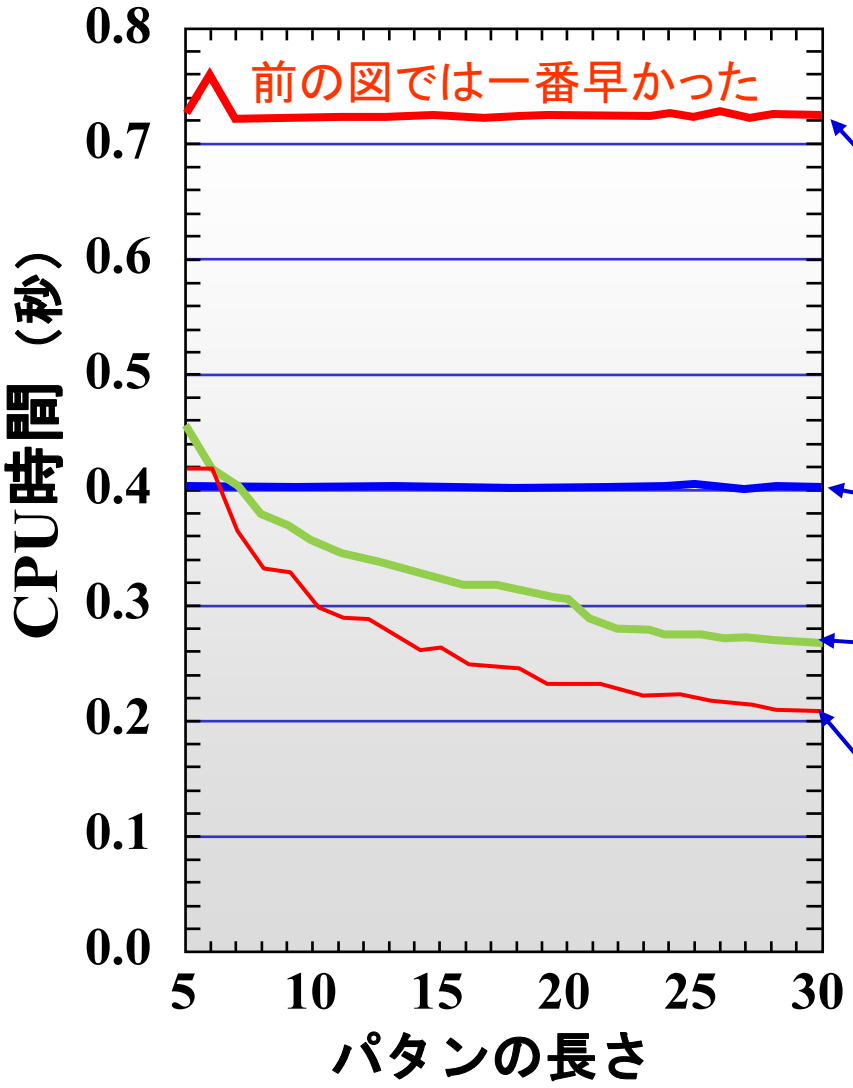
9

辞書

G → A B
H → D E
I → G C

サイズ:256
= 1 byte分

目標2の達成



AlphaStation XP1000
(Alpha21264: 667MHz)
Tru64 UNIX V4.0F

Medline（英文テキスト）
60.3Mbyte

非圧縮テキストを**KMP**で照合

「展開しないで」法

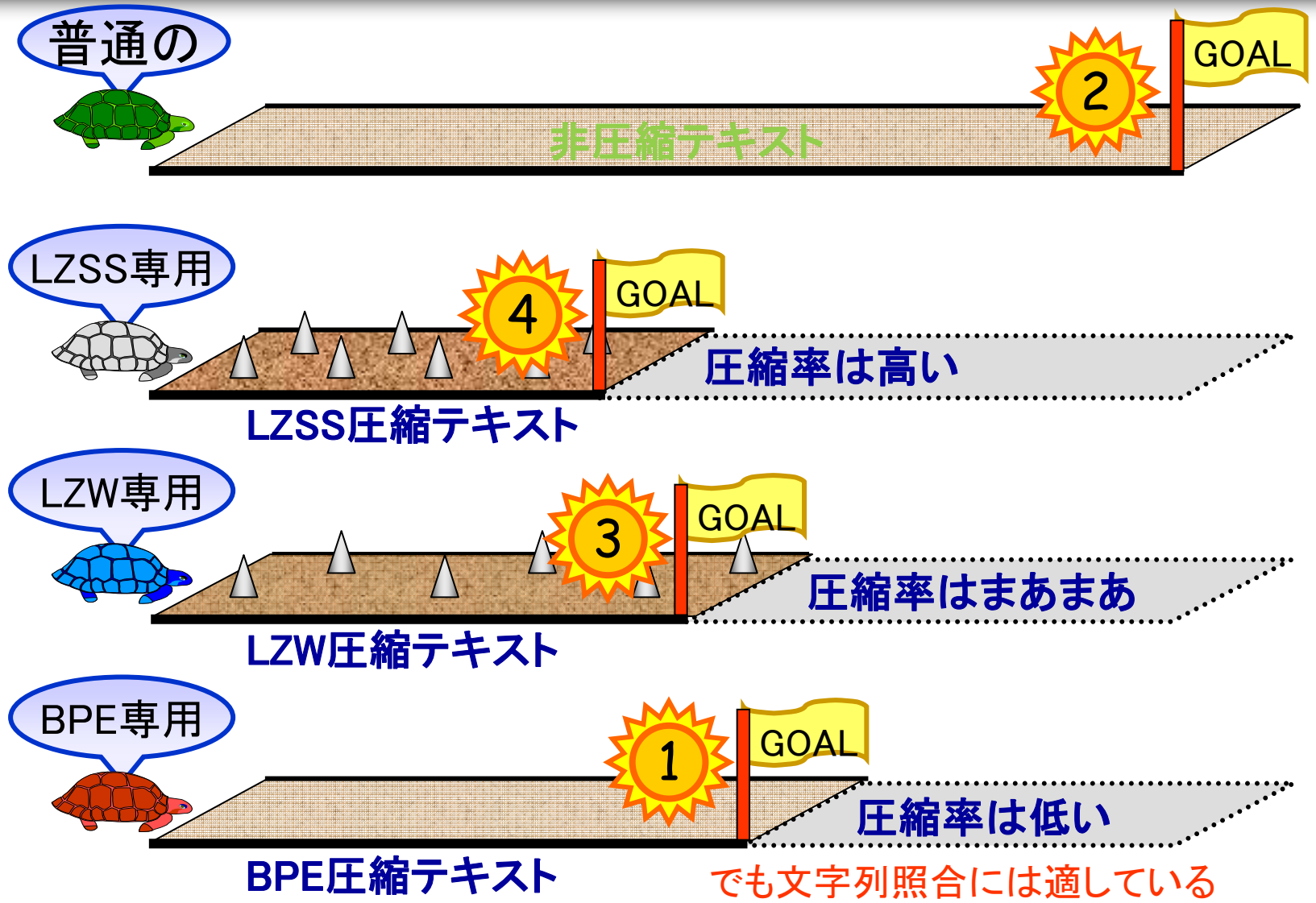
BPE圧縮テキストに対する照合(KMP)

非圧縮テキストを**Agrep**で照合

「展開しないで」法

BPE圧縮テキストに対する照合(BM)
Shibata, et al. (2000)

以上の話をまとめると…


普通の
非圧縮テキスト
2
GOAL
LZSS専用
4
GOAL
圧縮率は高い
LZSS圧縮テキスト
LZW専用
3
GOAL
圧縮率はまあまあ
LZW圧縮テキスト
BPE専用
1
GOAL
圧縮率は低い
BPE圧縮テキスト
でも文字列照合には適している

パラダイムシフト

圧縮技術を応用して
様々な処理の困難さの壁を
打ち破る

データを圧縮することで
文字列照合を高速化できる！

保存コストや転送コストを
低減するためにデータ圧縮技術を用いる

圧縮技術で･･･する

- 長大な二つの文字列どうしの類似度を《圧縮技術で》高速化する
  - “A Sub-quadratic Sequence Alignment Algorithm for Unrestricted Cost Matrices”,
    M. Crochemore, G. M. Landau, and M. Ziv-Ukelson, *Proceeding of 13th Symposium on Discrete Algorithm*, pp.679-688, 2002
- 巨大なグラフ構造を《圧縮技術で》オンメモリに載せて処理する
  - 中野眞一（群馬大学）「クエリサポート付きグラフ圧縮」
    極大平面グラフを2m+o(n) bitで、かつクエリサポート
- XML（半構造データ）に対する問い合わせを《圧縮技術で》高速化する
  - 舞田哲哉、坂本比呂志（九州工業大学）

その他の結果

- 文法圧縮テキスト上のパターン照合
  - S. Mitarai, et al., Compressed Pattern Matching for SEQUITUR, In *Proc. Data Compression Conference 2001*, pp. 469-478, IEEE Computer Society, 2001.
- 圧縮テキストに対する近似文字列照合
  - T. Matsumoto, T. Kida, et al., Bit-parallel approach to approximate string matching in compressed texts, In Proc. *7th International Symp. On String Processing and Information Retrieval*, pp.221-228, IEEE Computer Society, 2000.
  - G. Navarro, T. Kida, et al., Faster Approximate String Matching over Compressed Text, In *Proc. Data Compression Conference 2001*, pp. 459-468, IEEE Computer Society, 2001.
- BWT圧縮に関するパターン照合
  - A. Firth, T. Bell, A. Mukherjee, and D. Adjeroh: A comparison of BWT approaches to string pattern matching, *Software—Practice & Experience*, Vol.35(13), pp.1217-1258, 2005.

第6回 まとめ

- データ圧縮されたテキスト上でのパターン照合アルゴリズム
  - Huffman符号化テキストに対する照合 → 同期つきオートマトン
  - LZW圧縮テキストに対する照合 → KMP(AC)の遷移をLZW上で模倣
- 統一的枠組み: Collage system
  - 辞書式圧縮法によって圧縮されたテキストを統一的に表現するための形式的体系
  - どのような圧縮法がパターン照合に適しているかを明確にした
- 圧縮による照合の高速化という視点
  - BPE圧縮: 圧縮率は低いが、パターン照合を高速化する
  - もとのテキストに対してパターン照合するよりも高速に照合できた
- 大きなパラダイムシフト
  - データ圧縮技術は、単にデータを小さくするだけではない使い道がある

- 次回(「情報検索とパターン照合」最終回)のテーマ
  - これまでの講義でできなかったトピックスについて、いろいろ

Kida, et al.[1998]のアルゴリズムの擬似コード

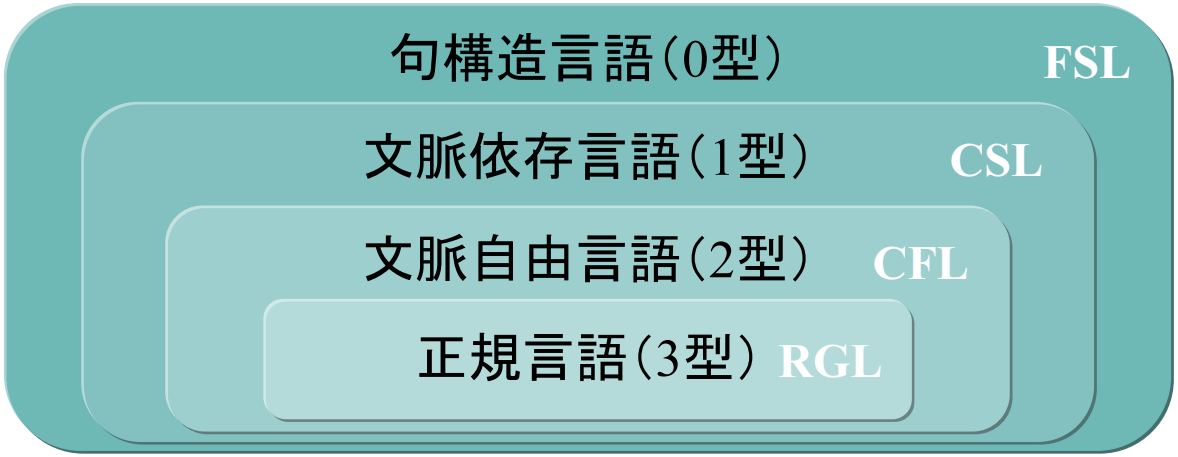
PMonLZW (E(T) = $u_1u_2\ldots u_n$, Π:パターン集合)

```
Construct AC machine and generalized suffix trie for Π;
Initialize the dictionary trie for E(T);
Preprocess Jump(q,u) and Output(q,u)
                    for any q and u∈{あるパターン π ∈ Π のfactor}
l ← 0;
q ← q0;
for i ← 1…n do
    for each ⟨d , π⟩∈Output(q, ui) do
        report pattern π occurs at position l+d;
    q ← Jump(q, ui);
    l ← l + |ui|;
    Update the dictionary trie;
      /* ノードui+1に対する文字列がDに登録される */
    Update variables for Jump(q, ui+1) and Output(q, ui+1);
      /* δ(ε ,ui+1) やA(ui+1)、ui+1'、|ui+1|等が親ノードを用いて計算される */
    end of for
end of for
```

付録：文脈自由文法について

- 4つ組**G=(N,Σ,P,S)**によって定義される。
  - N：非終端アルファベットと呼ばれる空でない有限集合。
  - Σ：終端アルファベットと呼ばれる空でない有限集合。
  - P：N×(N∪Σ)*の有限部分集合。
    Pの要素(A, α)は生成規則と呼ばれ、A→αと書かれる。
  - S：S∈Nで、開始記号という。
- BNF(Backus Naur Form)は文脈自由文法の表記法の一つ。

Chomsky階層の4言語族



付録：文脈自由文法の例（1）

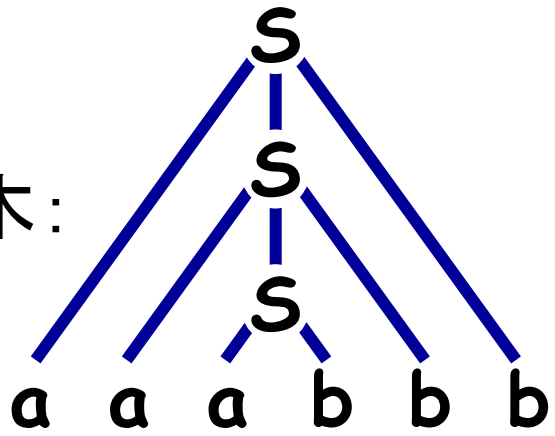
文脈自由文法$G=(N,\Sigma,P,S)$

$N=\{S\}$, $\Sigma=\{a,b\}$, $P=\{S\rightarrow ab,\ S\rightarrow aSb\}$

$$S \Rightarrow aSb \Rightarrow aaSbb \Rightarrow aaabbb$$

$L(G)=\{a^n b^n \mid n$ は 1以上の整数$\}$

構文木：



付録：文脈自由文法の例（2）

文脈自由文法$G=(N,\Sigma,P,S)$

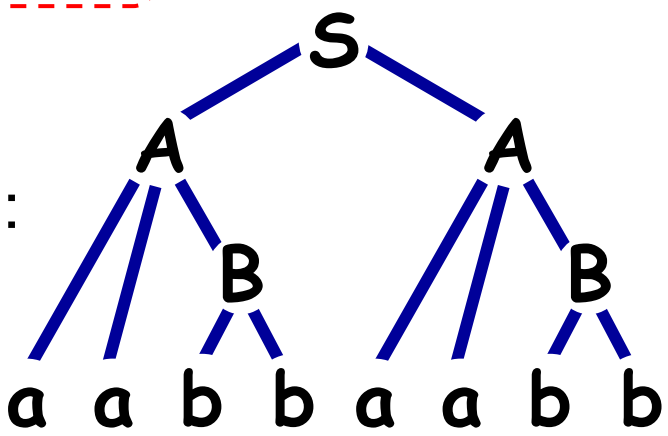
$N=\{S,A,B\}$, $\Sigma=\{a,b\}$,

$P=\{S\rightarrow AA,\ A\rightarrow aaB,\ B\rightarrow bb\}$

$$S \Rightarrow AA \Rightarrow aaBaaB \Rightarrow aabbaabb$$

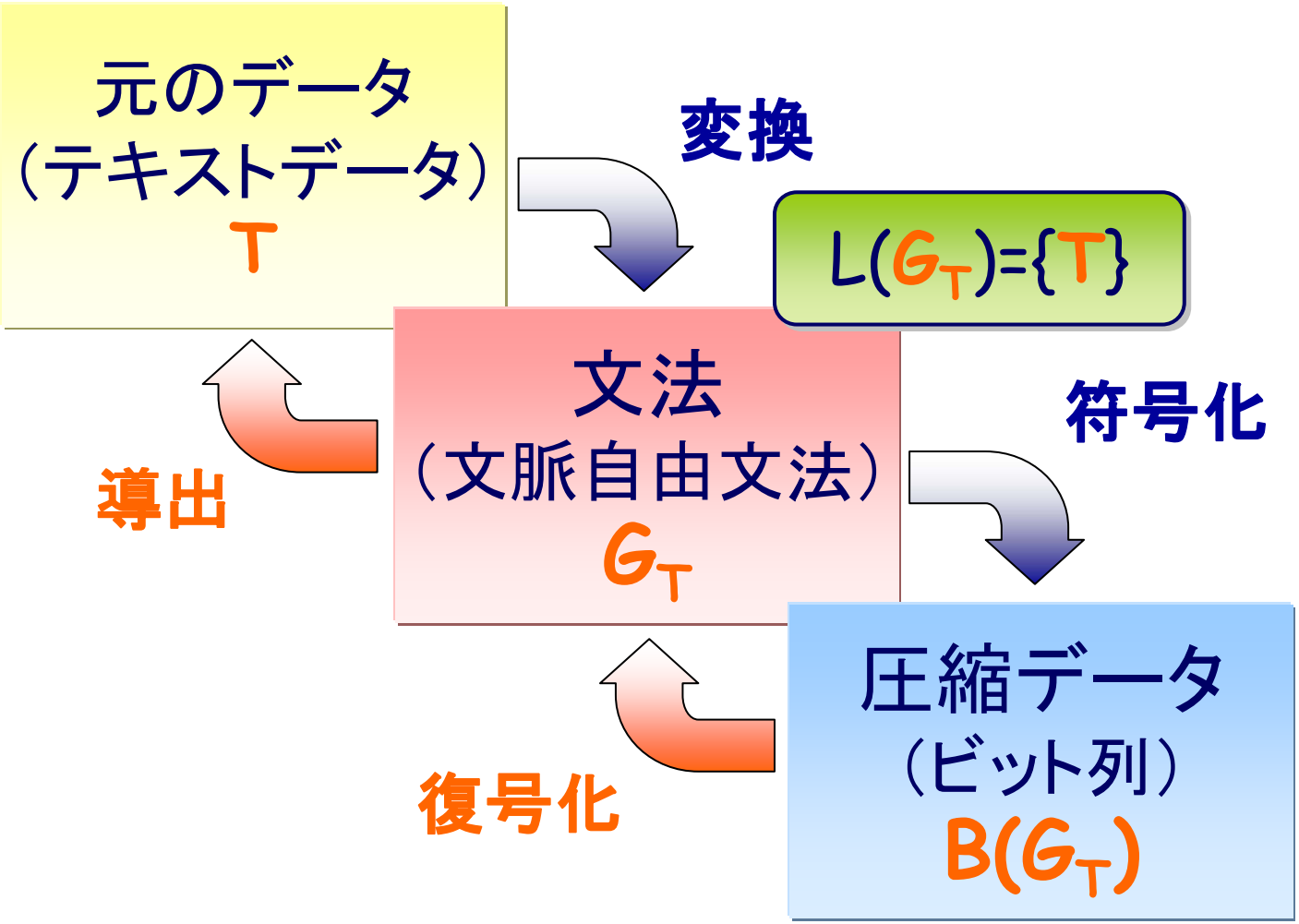
$L(G)=\{aabbaabb\}$

構文木：



付録：文法変換に基づく圧縮の考え方

John C. Kieffer and En-hui Yang, 2000.



付録：これまでの研究（C. Kieffer ら調べによる）

[G固定：構文木を圧縮]

- R. Cameron [1988]
- E. Kawaguchi and T. Endo [1980]
- E. Kourapova and B. Ryabko [1995]

[L(G)={x} となる G を圧縮]

- C. Cook, A. Rosenfeld, and A. Aronson [1976]
- J. Storer and T. Szymanski [1995]
- C. Nevill-Manning and I. Witten [1997]