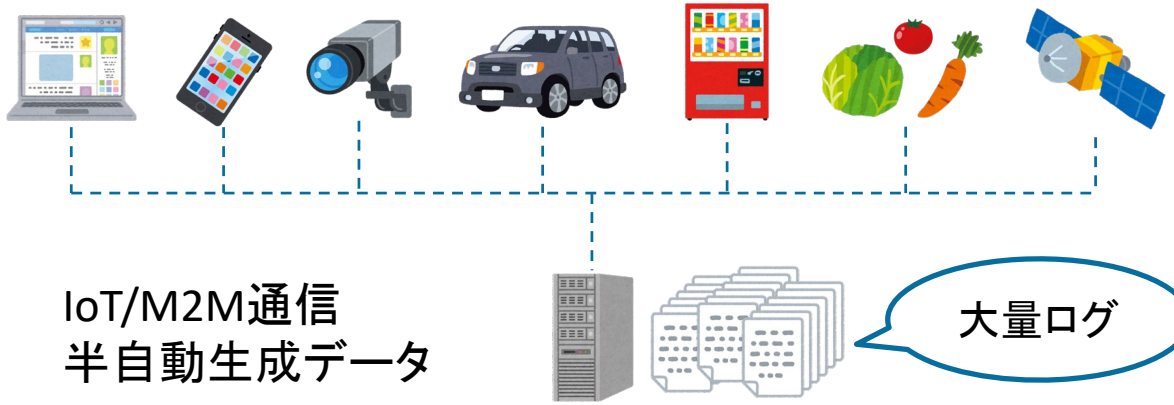
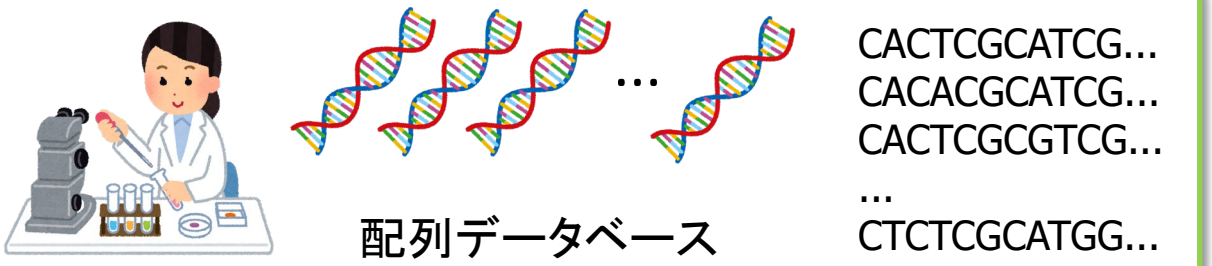
文字列圧縮アルゴリズムの感度

赤木亨　（九州大学）
舩越満　（九州大学）
○稲永俊介（九州大学）

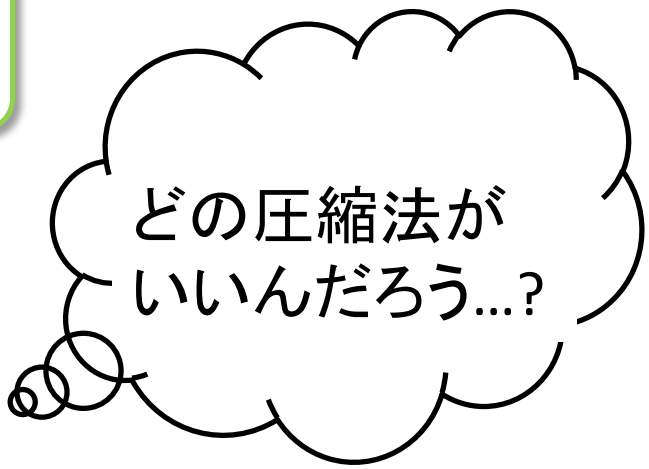
データ圧縮


IoT/M2M通信
半自動生成データ
大量ログ


CACTCGCATCG...
CACACGCATCG...
CACTCGCGTCG...
...
CTCTCGCATGG...
配列データベース


これ圧縮しといて
上司


zip? gzip? lha?
7z? bzip2? xz?
新入社員


どの圧縮法が
いいんだろう...?

圧縮法の新評価指標

圧縮ソフトウェア C のレビュー1

T = ababaaba…ab

$C(T)$

テストデータが相当縮みました！
圧縮解凍も爆速です！
しかも圧縮したまま検索できます！

圧縮ソフトウェア C のレビュー2

T' = a**X**abaaba..ab

$C(T')$

でも、試しに1文字編集してみたら
いきなり圧縮率悪くなりました…

レビュー読んでみたら
(1) 圧縮率
(2) 圧縮/展開速度
(3) 圧縮検索
全部揃った C が良さそう！

え～、そんなことあるの？

うちの会社で扱うデータは
更新されるから不安だなぁ…

本研究：新たな評価指標 **(4) 圧縮感度** を提案

圧縮感度

【定義】

圧縮アルゴリズム C の感度

T' は T に1文字編集操作（挿入・削除・置換）を行って得られる任意の文字列

最悪時感度（比）

$$\max\{|C(T')|/|C(T)| : T \in \Sigma^n, \mathrm{EditDistance}(T, T') = 1\}$$

最悪時感度（差分）

$$\max\{|C(T')|-|C(T)| : T \in \Sigma^n, \mathrm{EditDistance}(T, T') = 1\}$$

【直感的理解】

感度が小さい圧縮 ⇔ 編集やエラーに影響されにくい圧縮

編集やエラーを含むデータ

Wikipediaページの更新履歴

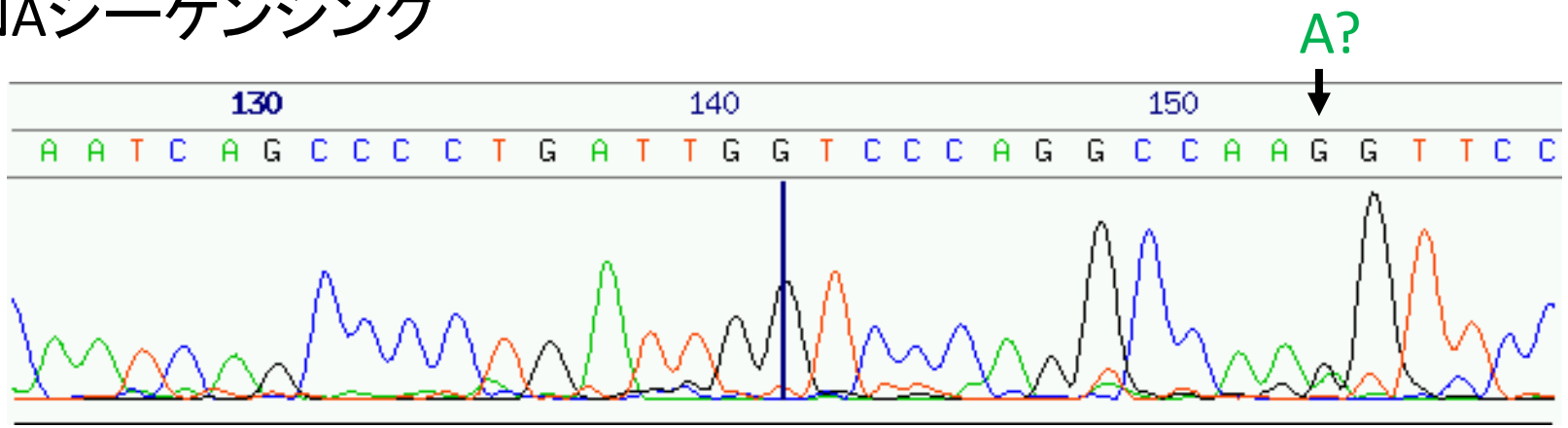
- ＊ [[bzip]] (~~bunzip~~) – 特許~~侵害~~のために配布が中止された高圧縮形式。[[算術符号]]使用。

+ ＊ [[bzip]] (*.bz) – 特許問題のために配布が中止された高圧縮形式。[[算術符号]]使用。

- ＊ [[bzip2]] (~~bunzip ver.2~~) – 主に[[UNIX]]で使われるオープンソースの高圧縮形式。bzipの後継として開発された。

+ ＊ [[bzip2]] (*.bz 2) – 主に[[UNIX]]で使われるオープンソースの高圧縮形式。bzipの後継として開発された。

DNAシーケンシング



http://www.chevreux.org/thesis/node10.html

圧縮感度(比)

圧縮法・反復性指標
LZ77 z
LZSS s
双方向スキーム b
最小文法 g^*
BISECTION g_{BISCTN}
GCIS g_{GCIS}
LZ78 g_{78}
RLBWT r
部分文字列複雑性 δ

圧縮感度(比)

関連研究はわずか
感度も良くない

圧縮法・反復性指標	上界	下界
LZ77 z		
LZSS s		
双方向スキーム b		
最小文法 g^*		
BISECTION g_{BISCTN}		
GCIS g_{GCIS}		
LZ78 g_{78}		$\Omega(n^{1/4})$ [Lagarde & Perifel SODA 2018]
RLBWT r		$\Omega(\log n)$ [SOFSEM 2021]
部分文字列複雑性 δ		

RLBWT の圧縮感度（下界）

r: 入力文字列のローテーションを辞書式順序に整列した行列の最右列文字列の**連長圧縮サイズ**

$\tilde{F}$: 逆向きフィボナッチ文字列

$\tilde{F}_6$	baabaababaaba
2	aabaababaabab
10	aababaabaabab
5	aababaababaab
13	abaabaababaab
8	abaababaabaab
3	abaababaababa
11	ababaabaababa
6	ababaababaaba
1	baabaababaaba
9	baababaabaaba
4	baababaababaa
12	babaabaababaa
7	babaababaabaa

辞書式順序にソート

b$\tilde{F}_6$	bbaabaababaaba
3	aabaababaababb
6	aababaababbaab
11	aababbaabaabab
4	abaababaababba
9	abaababbaabaab
7	ababaababbaaba
12	ababbaabaababa
14	abbaabaababaab
2	baabaababaabab
5	baababaababbaa
10	baababbaabaaba
8	babaababbaabaa
13	babbaabaababaa
1	bbaabaababaaba

b の追加によってソート順が大きく変わる

任意の $i \geq 2$ について，$r(\tilde{F}_i) = 2$ かつ $r(\mathrm{b}\tilde{F}_i) = i = \log_\phi n$

➔ RLBWT の圧縮感度の下界は $r(\mathrm{b}\tilde{F}_i)/r(\tilde{F}_i) = \Omega(\log n)$

圧縮感度(比)

他の圧縮アルゴリズムの
感度はどれくらいなんだろう…？

圧縮法・反復性指標	編集操作	上界	下界
LZ77 z	任意		
LZSS s	挿入		
	削除・置換		
双方向スキーム b	任意		
最小文法 g^*	任意		
BISECTION g_{BISCTN}	任意		
GCIS g_{GCIS}	任意		
LZ78 g_{78}	任意		$\Omega(n^{1/4})$ [Lagarde & Perifel SODA 2018]
RLBWT r	挿入		$\Omega(\log n)$ [SOFSEM 2021]
部分文字列複雑性 δ	削除		
	挿入・置換		

圧縮感度(比)(**本研究成果**)

主要な圧縮手法の感度は **O(1)** であることを証明

圧縮法・反復性指標	編集操作	上界	下界
LZ77 z	任意	**2**	**2**
LZSS s	挿入	**2**	**2**
	削除・置換	**3**	**3**
双方向スキーム b	任意	**2**	**2**
最小文法 g^*	任意	**2**	-
BISECTION g_{BISCTN}	任意	**2**	**2**
GCIS g_{GCIS}	任意	**4**	**4**
LZ78 g_{78}	任意	-	$\Omega(n^{1/4})$ [Lagarde & Perifel SODA 2018]
RLBWT r	挿入	**$O(\log r \log n)$**	$\Omega(\log n)$ [SOFSEM 2021]
部分文字列複雑性 δ	削除	**1.5**	**1.5**
	挿入・置換	**2**	**2**

圧縮感度(比)(**本研究成果**)

多くの圧縮法について
タイトな上下界を与えた

圧縮法・反復性指標	編集操作	上界	下界	
LZ77 z	任意	**2**	**2**	タイト!
LZSS s	挿入	**2**	**2**	タイト!
	削除・置換	**3**	**3**	タイト!
双方向スキーム b	任意	**2**	**2**	タイト!
最小文法 g^*	任意	**2**	-	
BISECTION g_{BISCTN}	任意	**2**	**2**	タイト!
GCIS g_{GCIS}	任意	**4**	**4**	タイト!
LZ78 g_{78}	任意	-	$\Omega(n^{1/4})$	[Lagarde & Perifel SODA 2018]
RLBWT r	挿入	$\mathbf{O(\log r \log n)}$	$\Omega(\log n)$	[SOFSEM 2021]
部分文字列複雑性 δ	削除	**1.5**	**1.5**	タイト!
	挿入・置換	**2**	**2**	タイト!

圧縮感度(比)(**本研究成果**)

LZSS圧縮について説明

圧縮法・反復性指標	編集操作	上界	下界	
LZ77 z	任意	**2**	**2**	タイト!
LZSS s	挿入	**2**	**2**	タイト!
	削除・置換	**3**	**3**	タイト!
双方向スキーム b	任意	**2**	**2**	タイト!
最小文法 g^*	任意	**2**	-	
BISECTION g_{BISCTN}	任意	**2**	**2**	タイト!
GCIS g_{GCIS}	任意	**4**	**4**	タイト!
LZ78 g_{78}	任意	-	$\Omega(n^{1/4})$	[Lagarde & Perifel SODA 2018]
RLBWT r	挿入	$\mathbf{O(\log r \log n)}$	$\Omega(\log n)$	[SOFSEM 2021]
部分文字列複雑性 δ	削除	**1.5**	**1.5**	タイト!
	挿入・置換	**2**	**2**	タイト!

LZSS圧縮 (zip / gzip / lha / 7zip / png …)

LZSS [Storer-Szymanski, 1982]

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$T =$	a	b	a	a	b	a	b	a	b	a	b	a	b	a	b

文字列を左から右にスキャンしながら分割していく

- 初出現の文字が出たら区切る
- 直前の分割位置までに出現している
  最長の部分文字列で区切る

LZSS圧縮 (zip / gzip / lha / 7zip / png …)

LZSS [Storer-Szymanski, 1982]

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$T =$	a	b	a	a	b	a	b	a	b	a	b	a	b	a	b

初出現の文字

文字列を左から右にスキャンしながら分割していく

- 初出現の文字が出たら区切る
- 直前の分割位置までに出現している
  最長の部分文字列で区切る

LZSS圧縮 (zip / gzip / lha / 7zip / png …)

LZSS [Storer-Szymanski, 1982]

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$T =$	a	b	a	a	b	a	b	a	b	a	b	a	b	a	b

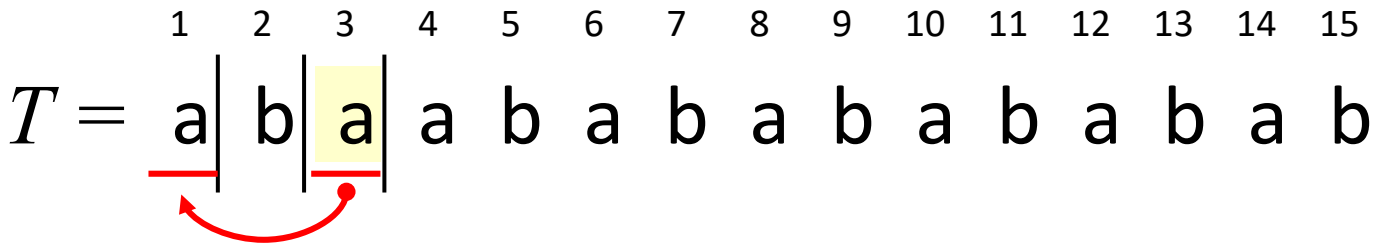
初出現の文字

文字列を左から右にスキャンしながら分割していく

- 初出現の文字が出たら区切る
- 直前の分割位置までに出現している
  最長の部分文字列で区切る

LZSS圧縮 (zip / gzip / lha / 7zip / png …)

LZSS [Storer-Szymanski, 1982]

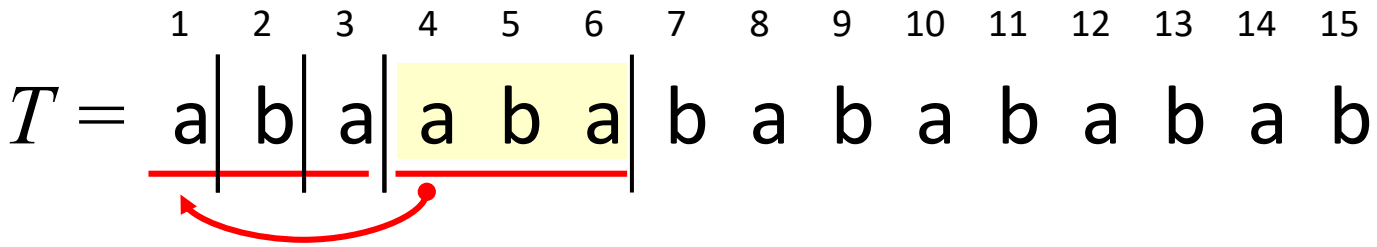


文字列を左から右にスキャンしながら分割していく

- 初出現の文字が出たら区切る
- 直前の分割位置までに出現している
  最長の部分文字列で区切る

LZSS圧縮 (zip / gzip / lha / 7zip / png …)

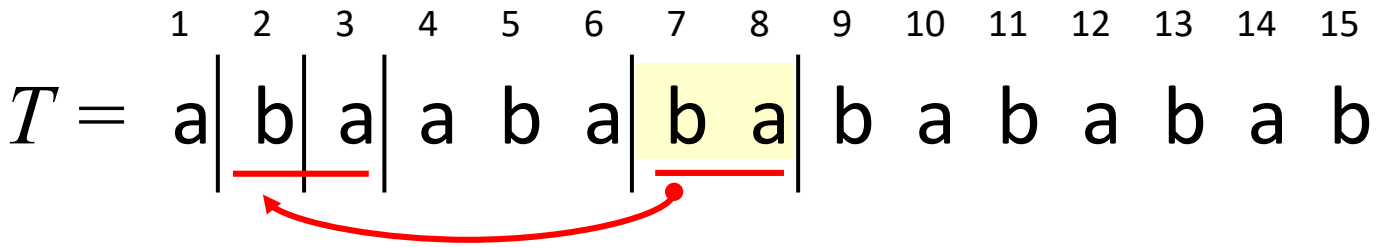
LZSS [Storer-Szymanski, 1982]



直前の分割位置までに出現している
最長の部分文字列で区切る

LZSS圧縮 (zip / gzip / lha / 7zip / png …)

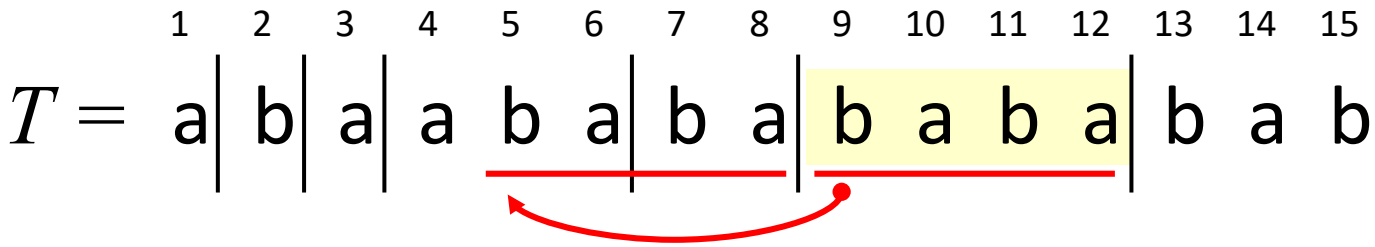
LZSS [Storer-Szymanski, 1982]



直前の分割位置までに出現している
最長の部分文字列で区切る

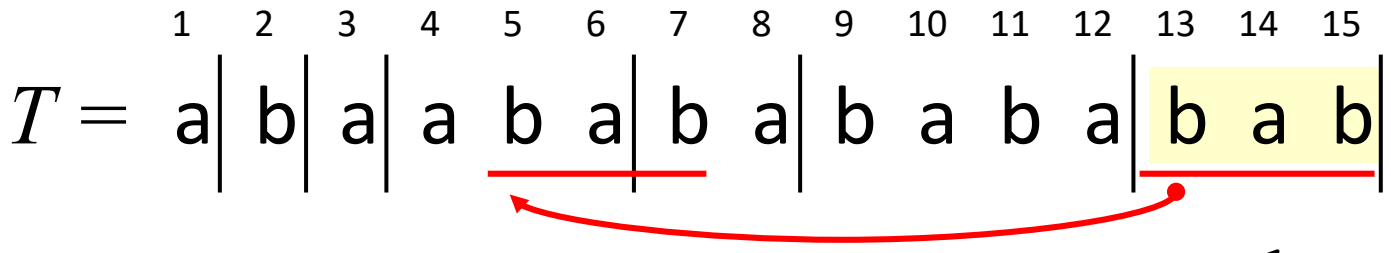
LZSS圧縮 (zip / gzip / lha / 7zip / png …)

LZSS [Storer-Szymanski, 1982]



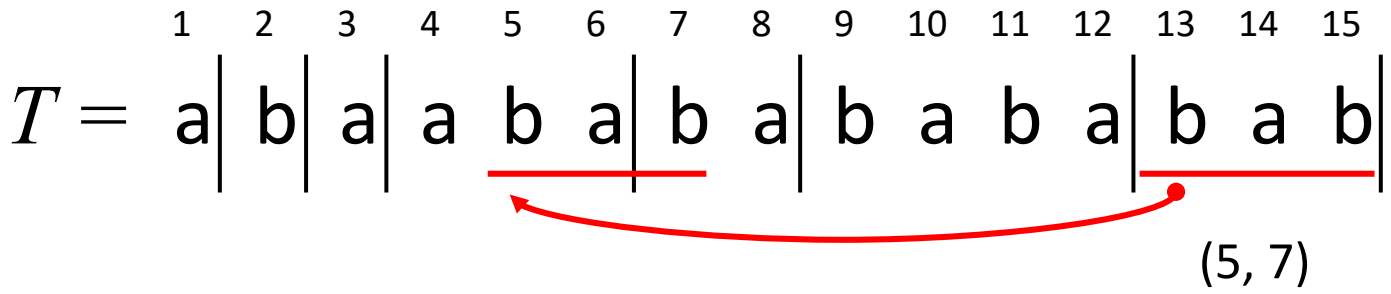
LZSS圧縮 (zip / gzip / lha / 7zip / png …)

LZSS [Storer-Szymanski, 1982]



LZSS圧縮 (zip / gzip / lha / 7zip / png …)

LZSS [Storer-Szymanski, 1982]



LZSS圧縮 (zip / gzip / lha / 7zip / png …)

LZSS [Storer-Szymanski, 1982]

$$T =$$

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
a	b	a	a	b	a	b	a	b	a	b	a	b	a	b

(a) (b) (1, 1) (1, 3) (2, 3) (5, 8) (5, 7)

圧縮サイズ（項数）＝ 7

LZSS圧縮 (zip / gzip / lha / 7zip / png …)

LZSS [Storer-Szymanski, 1982]

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$T =$	a	b	a	a	b	a	b	a	b	a	b	a	b	a	b
	(a)	(b)	(1, 1)	(1, 3)			(2, 3)			(5, 8)				(5, 7)	

圧縮サイズ（項数）＝ 7

LZSS（自己参照あり）

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$T =$	a	b	a	a	b	a	b	a	b	a	b	a	b	a	b
	(a)	(b)	(1, 1)	(1, 3)			(5, 13)								

圧縮サイズ（項数）＝ 5

LZSS圧縮の感度(上界)

s :編集前の文字列のLZSS圧縮サイズ
s':編集後の文字列のLZSS圧縮サイズ

定理 [LZSS圧縮の感度の上界]

任意の文字列に対して以下が成り立つ.

感度(比)　$s'/s \leq 3$

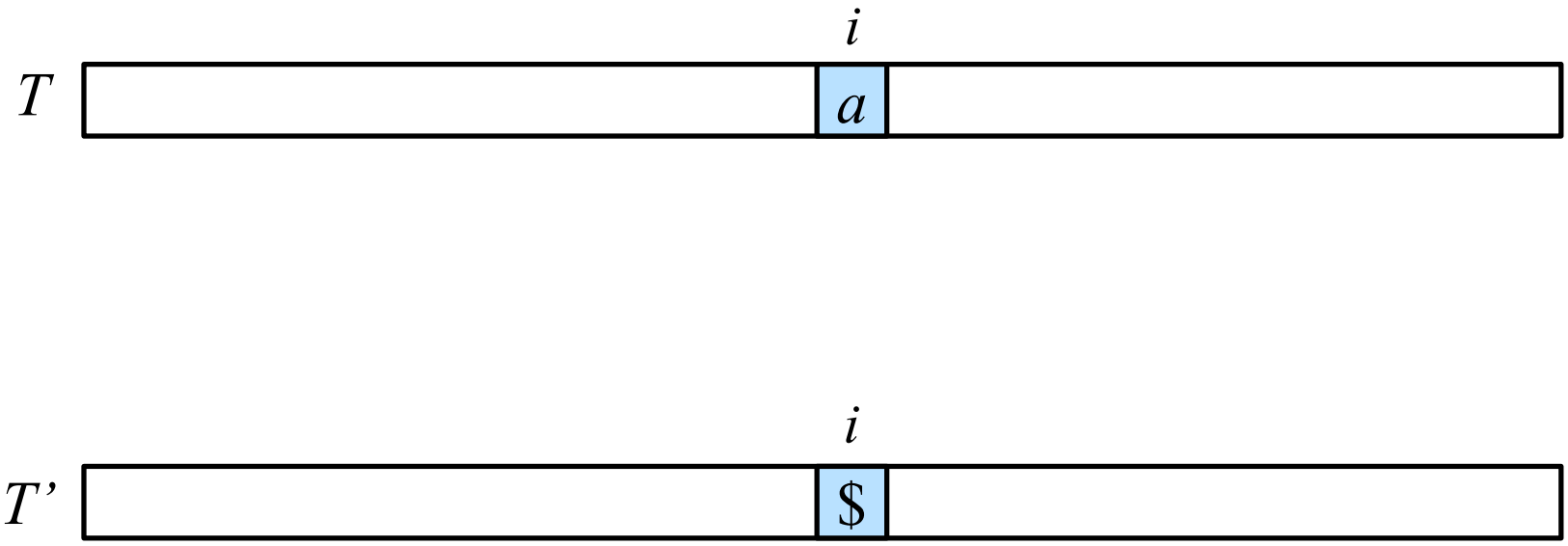
感度(差分)　$s' - s \leq 2s - 2$

※ この上界は自己参照なし/ありの両方で成立

LZSS圧縮の感度（上界）

【$s' \leq 3s$ を示す】

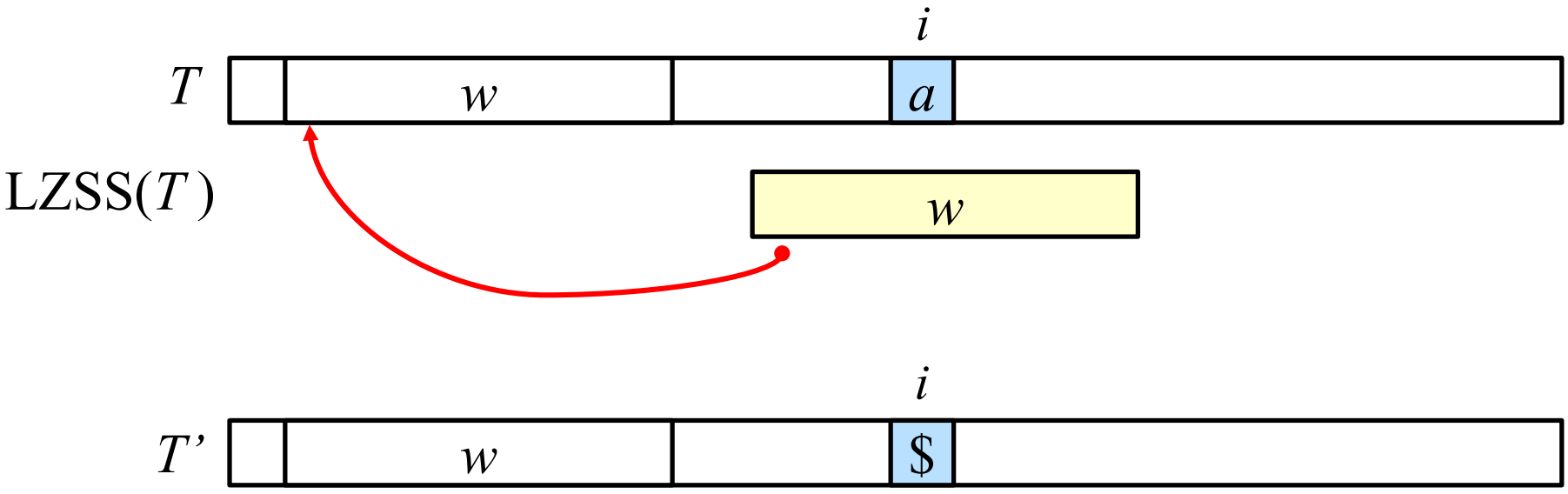
Claim 1: 編集位置 i を含む項 w について,
w の区間内から始まる項の個数は高々3に増える



LZSS圧縮の感度（上界）

【$s' \leq 3s$ を示す】

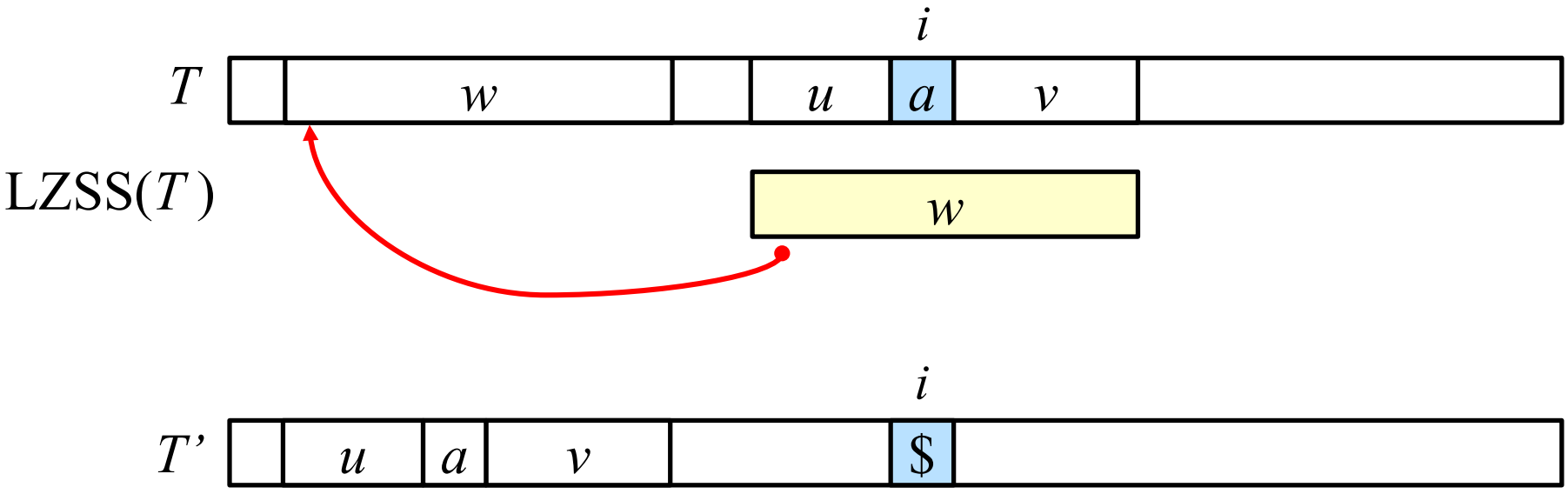
Claim 1: 編集位置 i を含む項 w について，
w の区間内から始まる項の個数は高々3に増える



LZSS圧縮の感度（上界）

【$s' \leq 3s$ を示す】

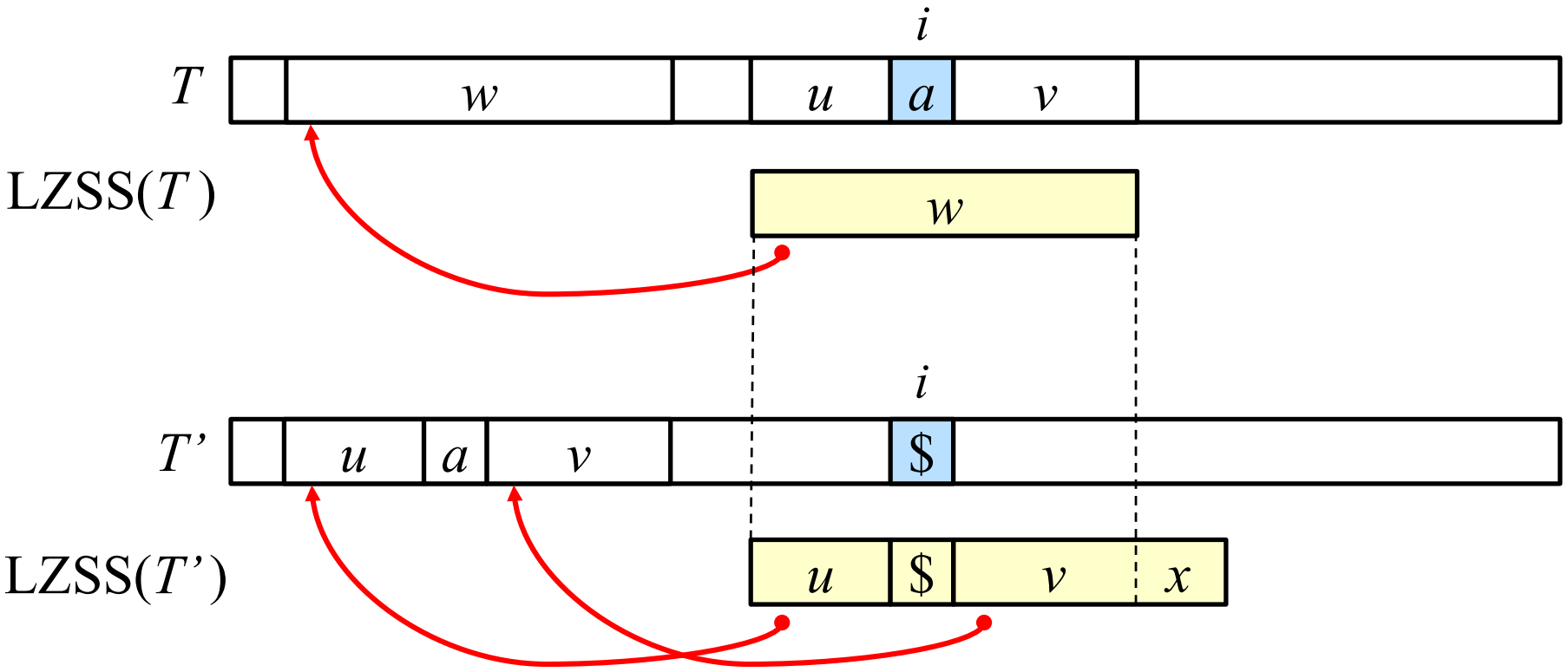
Claim 1: 編集位置 i を含む項 w について，
w の区間内から始まる項の個数は高々3に増える



LZSS圧縮の感度(上界)

【$s' \leq 3s$ を示す】

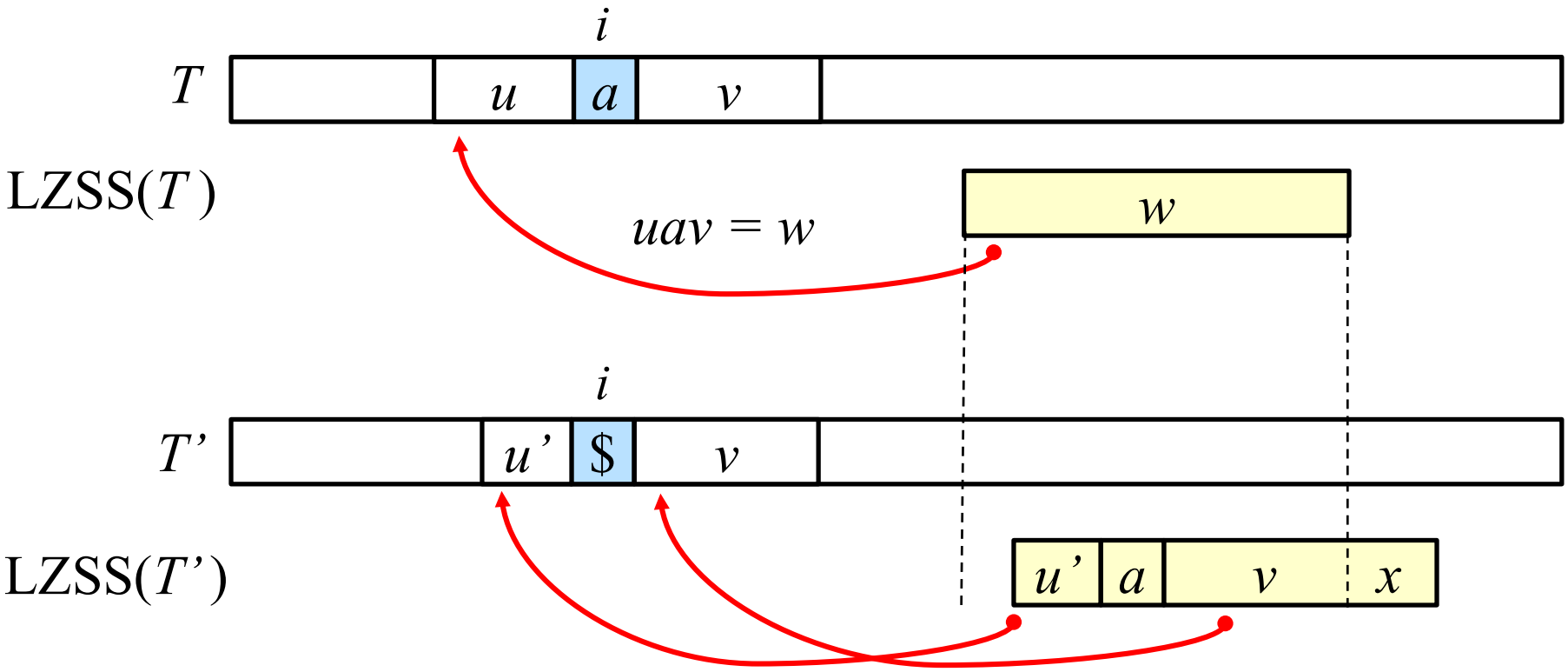
Claim 1: 編集位置 i を含む項 w について,
w の区間内から始まる項の個数は高々3に増える



LZSS圧縮の感度(上界)

【$s' \leq 3s$ を示す】

Claim 2: 編集位置 i より右側にある項 w について,
w の区間内から始まる項の個数は高々3に増える



LZSS圧縮の感度(上界・下界)

定理 [LZSS圧縮の感度の上界]

任意の文字列に対して以下が成り立つ.

感度(比)　$s'/s \leq 3$

感度(差分)　$s' - s \leq 2s - 2$

定理 [LZSS圧縮の感度の下界]

以下を満たす長さ n の文字列が存在する.

感度(比)　$\liminf_{s\to\infty}(s'/s) = 3$

感度(差分)　$s' - s = 2s - \Theta(\sqrt{s}) = \Omega(\sqrt{n})$

【下界の証明の概要】

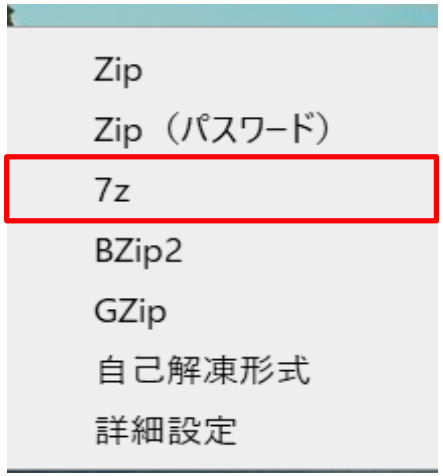
前ページの上界の証明に限りなく近い状況を満たす文字列の構成法を示した.

実際の圧縮ソフトウェアの感度

理論的に2倍とか3倍になることはわかったけど、
実際のソフトでそんな状況起こるのかな…？

圧縮ソフト 7ZIP の圧縮感度を調査

- 拡張子 .7z の圧縮形式をサポート
- 7z は zip の上位互換版
  （7z のほうが 30～70% 圧縮率がよい）
- 7z = LZSS＋RangeCoder



実際の圧縮ソフトウェアの感度

7z の感度が高くなるように
巧妙に構成された
とてもイジワルな文字列

stringX : 長さ 1,172,887 (1.12 MB)

!“#$%&‘()*+,./0123456789:;⇔?@ABCDEFGHIJKLMN!”#$%&’()*+,./01234
56789:;⇔?@ABCDEFGHIJKLM!“#$%&‘()*+,./0123456789:;<=>?@ABCDEF
GHIJKL!”#$%&’()*+,./0123456789:;<=>?@ ABCDE　　中略　　’!“#$%&!
”#$%!“#$!”#!“!!{OOPOPQOPQROPQRSOPQRSTOPQRSTUOPQRST 以降略

1083 番目の文字 “ { “ を “ ~ ” で置換

stringXp

!“#$%&‘()*+,./0123456789:;⇔?@ABCDEFGHIJKLMN!”#$%&’()*+,./01234
56789:;⇔?@ABCDEFGHIJKLM!“#$%&‘()*+,./0123456789:;<=>?@ABCDEF
GHIJKL!”#$%&’()*+,./0123456789:;<=>?@ABCDE　　中略　　’!“#$%&!
”#$%!“#$!”#!“!!~OOPOPQOPQROPQRSOPQRSTOPQRSTUOPQRST 以降略

7-Zipで圧縮したサイズ	stringX.7z	4,884 バイト
	stgingXp.7z	7,189 バイト

感度（比）
約1.5倍!!

まとめと今後の課題

- 圧縮アルゴリズムの新たな評価指標 **圧縮感度** を提案し，多くの圧縮アルゴリズムの感度の**タイトな上下界**を示した．
- **実際の圧縮ソフト**に対して，感度が約1.5倍になる実例を発見した（1文字/120万文字 の編集で達成）．

◆ 以下の圧縮法の上界・下界の解明

圧縮法・反復性指標	上界（比）	下界（比）
最小文法 g^*	2	-
LZ78 g_{78}	-	$\Omega(n^{1/4})$
RLBWT r	$O(\log r \log n)$	$\Omega(\log n)$
文字列アトラクタ γ	$O(\log n)$	2
AVL / a-balanced / Recomression 文法	$O(\log n)$	2
RePair / Greedy / Longest 文法	-	-

圧縮感度（差分）

圧縮法・反復性指標	上界	下界	下界 (nの関数)
LZ77圧縮 z	$z-1$	$z-1$	$\Omega(\sqrt{n})$
LZSS圧縮 s	$2s-s$	$2s-\Theta(\sqrt{s})$	$\Omega(\sqrt{n})$
AVL 文法	$O(s \log n)$	-	-
双方向スキーム b	$b+2$	$b/2-3$	$\Omega(\sqrt{n})$
最小文法 g^*	g^*	-	-
BISECTION g_{BISCTN}	g_{BISCTN}	$g_{\mathrm{BISCTN}}-4$	$2\log_2 n-4$
GCIS圧縮 g_{GCIS}	$3g_{\mathrm{GCIS}}$	$3g_{\mathrm{GCIS}}-29$	$3n/4+1$
LZ78圧縮 g_{78}	-	$\Omega({g_{78}}^{3/2})$	$\Omega(n/\log n)$
連長BW変換 r	$O(r \log r \log n)$	-	$\Omega(\log n)$
部分文字列複雑性 δ	1	1	1
文字列アトラクタ γ	$O(\gamma \log n)$	γ–3	$\Omega(\sqrt{n})$