第5回データ工学と情報マネジメントに関するフォーラム (DEIM2013), 福島, 2013年3月.

大規模テキストに対する共有辞書を用いた Re-Pair 圧縮法

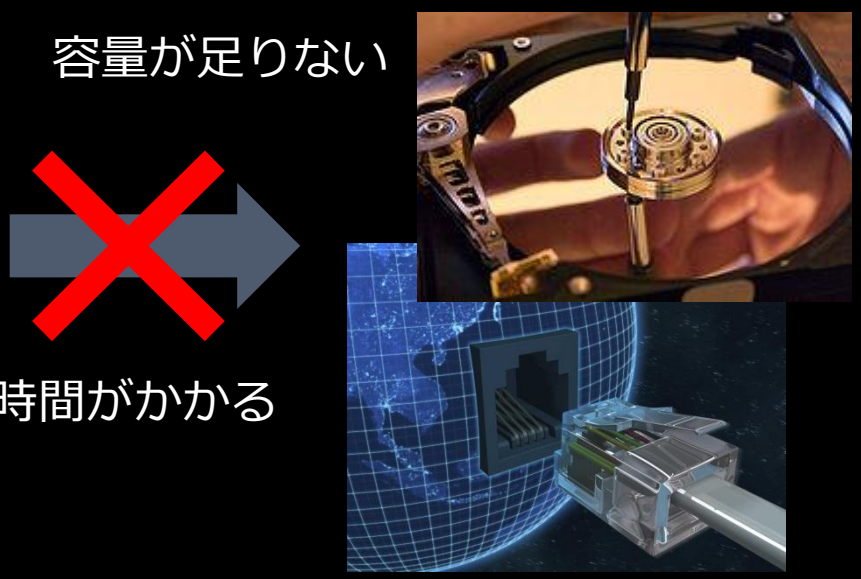
Variable-to-Fixed-Length Encoding for Large Texts Using Re-Pair Algorithm with Efficient Shared Dictionaries

◎ 関根 渓[†], 笹川 裕人[†], 吉田 諭史[†], 喜田 拓也[†]

[†]北海道大学大学院情報科学研究科

背景：巨大なデータ

- 計算機上で扱うデータの巨大化.



- 効率の良い圧縮手法の提案が望まれている.

背景：文法圧縮

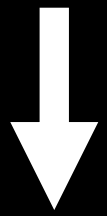
- 近年，文法圧縮に注目が集まっている.
  - 入力テキストデータを一意に生成する文脈自由文法を構築し，その文法を符号化する圧縮手法.
  - 良い圧縮率を達成出来る.

- 代表的な文法圧縮アルゴリズム
  - Re-Pair [Larsson and Moffat 1999]
  - SEQUITUR [Nevill-Manning et al. 1994]
  - BPE [Gage 1994]

Re-Pair アルゴリズム

- 最頻出の2-gramを新しい記号で置き換えていく.

ENOOOBOEOOOBEEOOOB



ENFOBOEFOBEEFOB

F → OO

Dictionary

Re-Pair アルゴリズム

- 全ての2-gramがユニークになったら変換終了

T: ENOOOBOEOOOBEEOOOB

ENFOBOEFOBEEFOB

ENABOEABEEAB

ENCOECEEC

ENCODED

圧縮

解凍

Dictionary

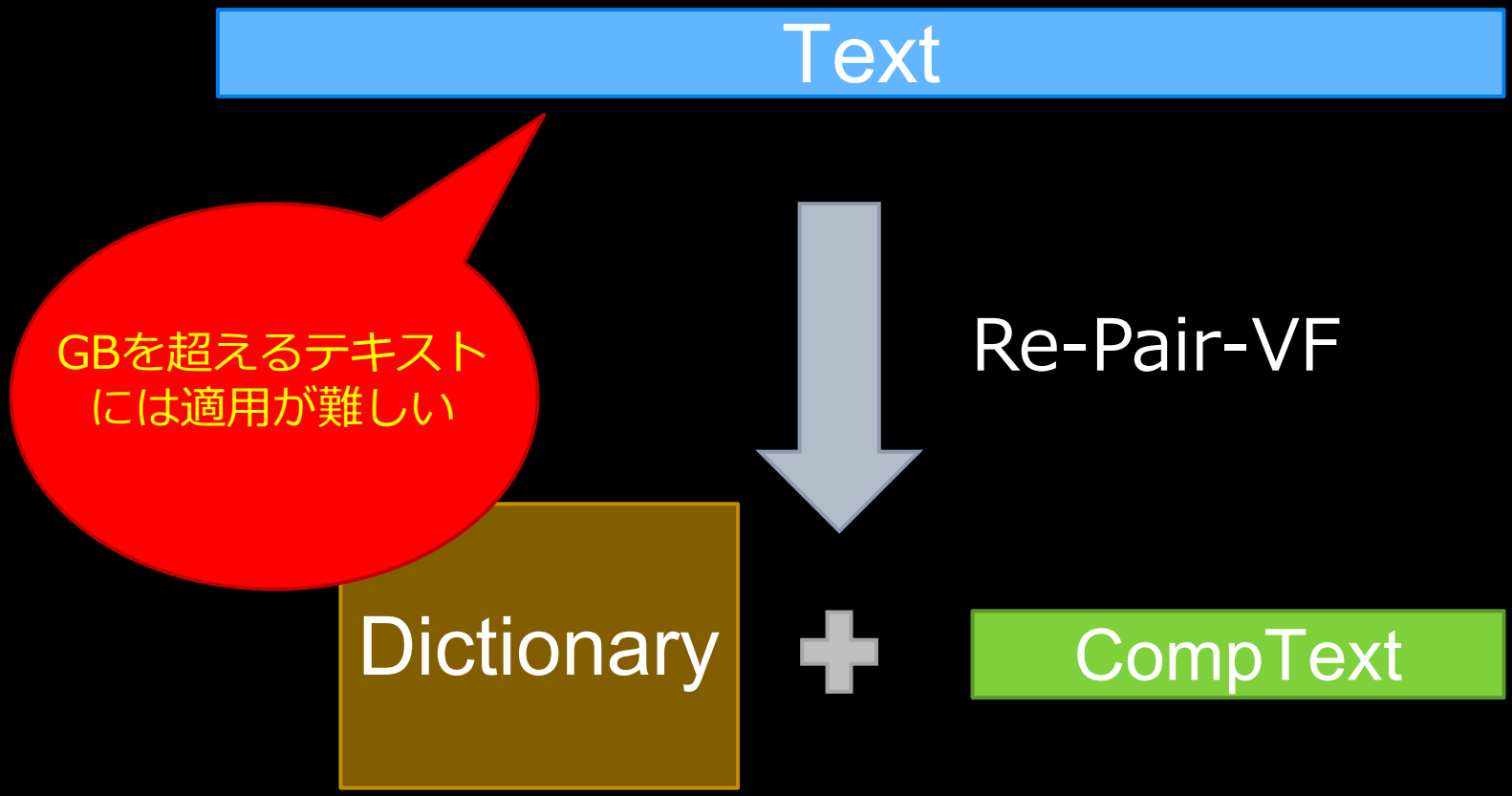
F → OO
A → FO
C → AB
D → EC

適当な2進符号化

Re-Pair-VF [Yoshida & Kida 2013]

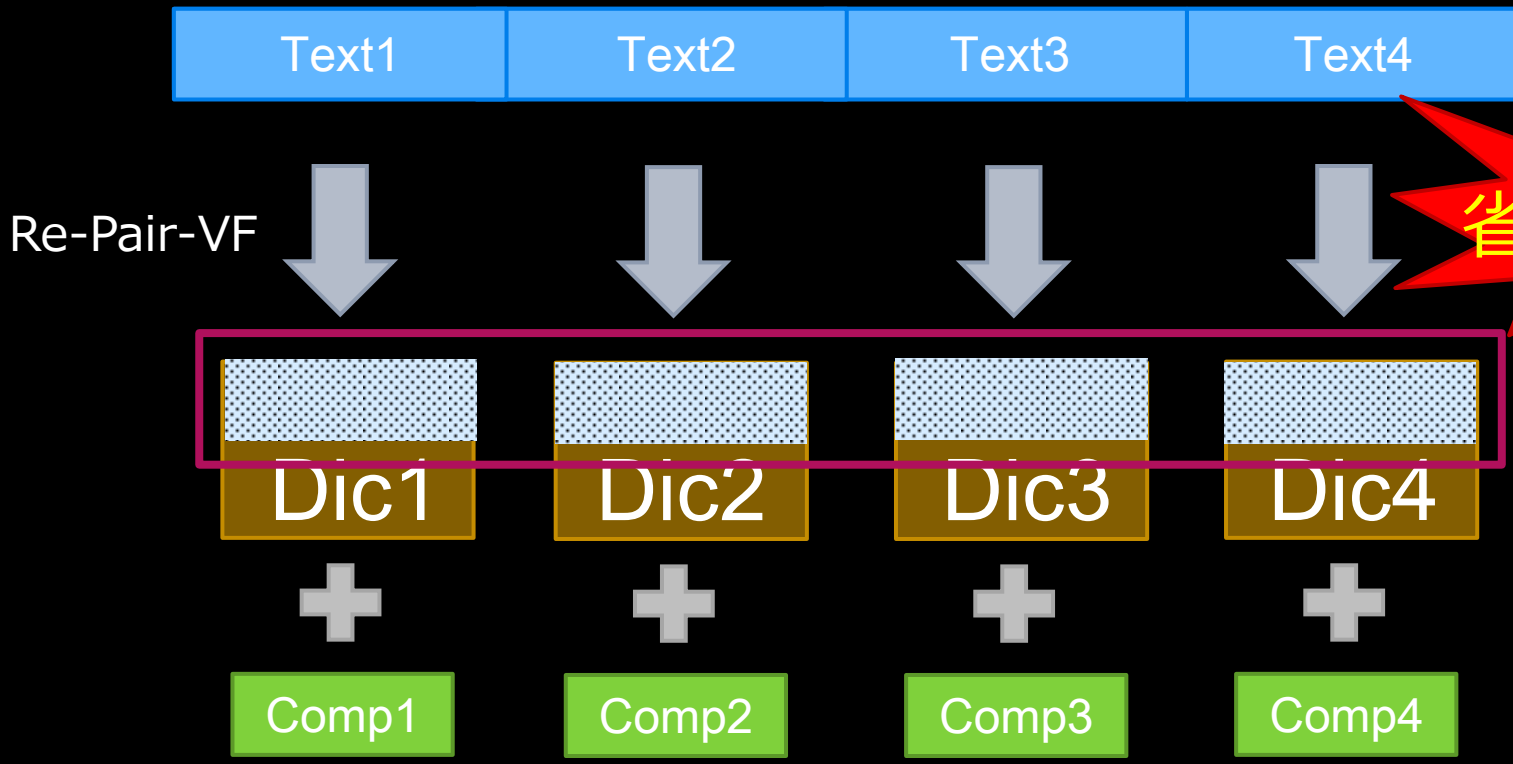
- 利点
  - 固定長符号化を用いつつも，十分によい圧縮率を達成. （gzipを超える）
  - 圧縮テキストが扱いやすい.
    例：圧縮パターン検索が高速
- 欠点
  - メモリ消費が激しい. （平均して入力テキストの10~20倍）

Re-Pair-VF アルゴリズム



対策：テキストのブロック化

テキストを分割
（ブロック化）



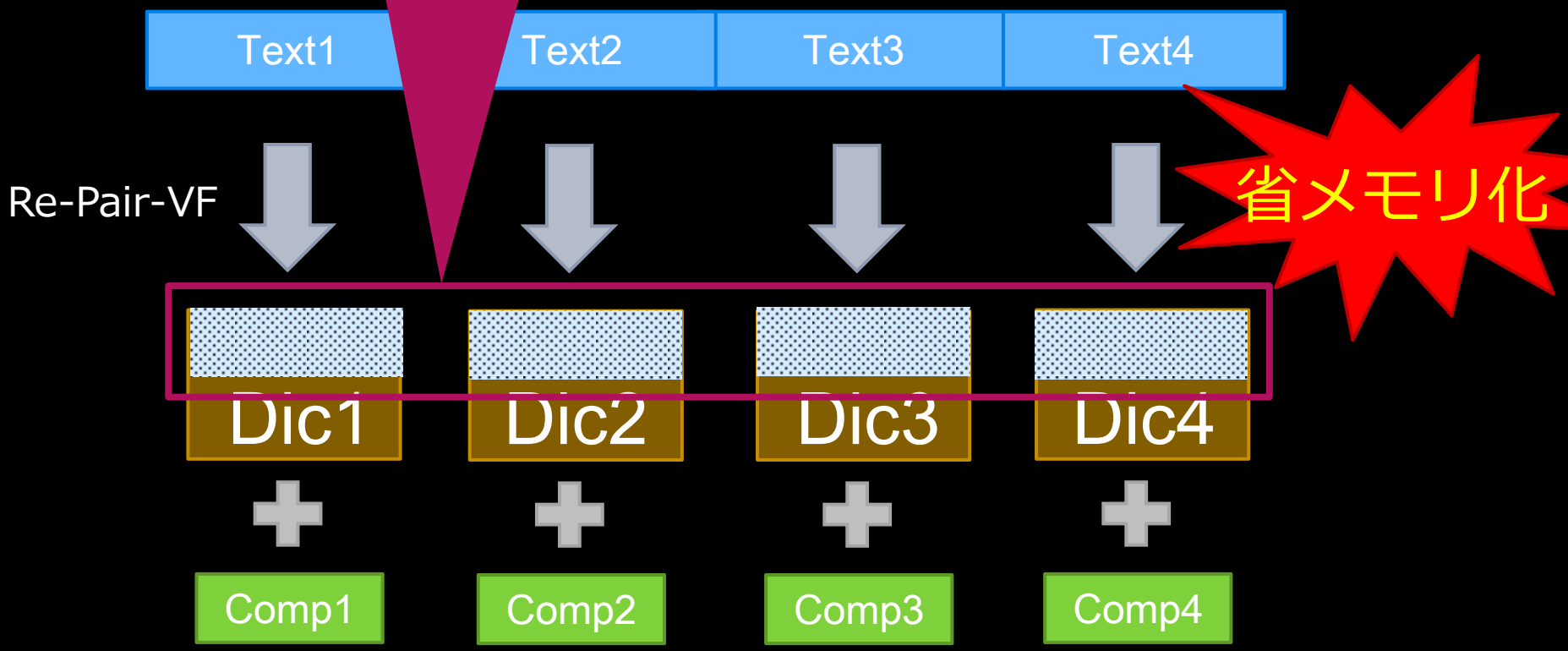
省メモリ化

出力される辞書の一部が同じエントリを持ってしまう．
（予備実験では，各ブロックで30%程度の2-gramの重複を確認）

のブロック化

冗長部分を共有化したい

テキストを分割
（ブロック化）



出力される辞書の一部が同じエントリを持ってしまう．
（予備実験では，各ブロックで30%程度の2-gramの重複を確認）

関連研究

- Re-Merge [Wan & Moffat. ‘07]
  - 各ブロックでRe-Pairを実行後，ブロック間で辞書のマージを行う.
  - 圧縮率は良い（英文テキストにおいて20%弱）が時間がかかる.

- Blocked-Re-Pair-VF [Sekine, Sasakawa et al. DBS ‘12]
  - 先頭ブロックのテキストから共有辞書を生成.
  - 圧縮率を悪化させることなく省メモリ化に成功.

Blocked-Re-Pair-VF の問題点

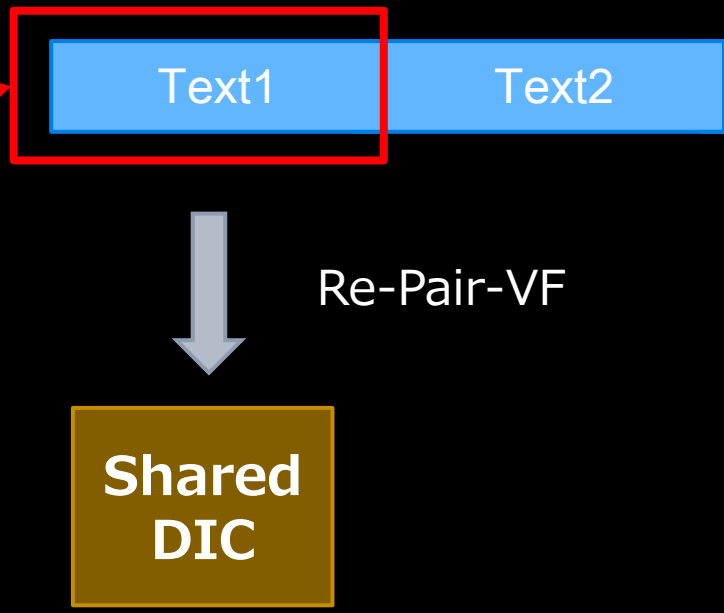
- Blocked-Re-Pair-VF [Sekine, Sasakawa et al. 2012]
  テキストの先頭が全体の文脈を内包していると仮定し,
  先頭ブロックのテキストから共有辞書を生成.

hoge.txt

Bob laughed so hard that the salmon carpaccio came out of his nose. The department manager tried to tackle the Santa Claus while completely nak...

```
__人人人人人人__
＞　突然のDNA　＜
￣Y^Y^Y^Y^Y￣
```

...GATA
CTAAACCCTAAAACCCCTTTTTTGAT
ACCCCAAATAGAAAAGGGTCCGTAA
AAATCACCATAATGATACCTGATTTT



RI[illegible]-VF の問題点

テキストの途中で
大きく文脈が変わる場合,
圧縮率悪化の可能性

[illegible] [Sekine, Sasakawa et al. 2012]

[illegible]全体の文脈を内包していると仮定し,

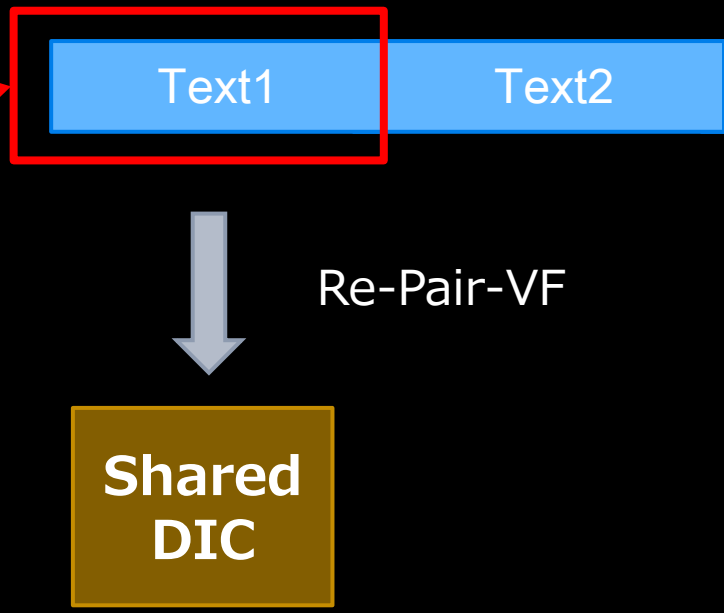
[illegible]ストから共有辞書を生成.

hoge.txt

Bob laughed so hard that the salmon carpaccio came out of his nose. The department manager tried to tackle the Santa Claus while completely nak...

```
  __人人人人人人__
  ＞　突然のDNA　＜
  ￣Y^Y^Y^Y^Y￣
```

...GATA
CTAAACCCTAAAACCCCTTTTTTGAT
ACCCCAAATAGAAAAGGGTCCGTAA
AAATCACCATAATGATACCTGATTTT



変則的なテキストにも柔軟に対応可能な圧縮アルゴリズムが必要

研究目的と主結果

目的

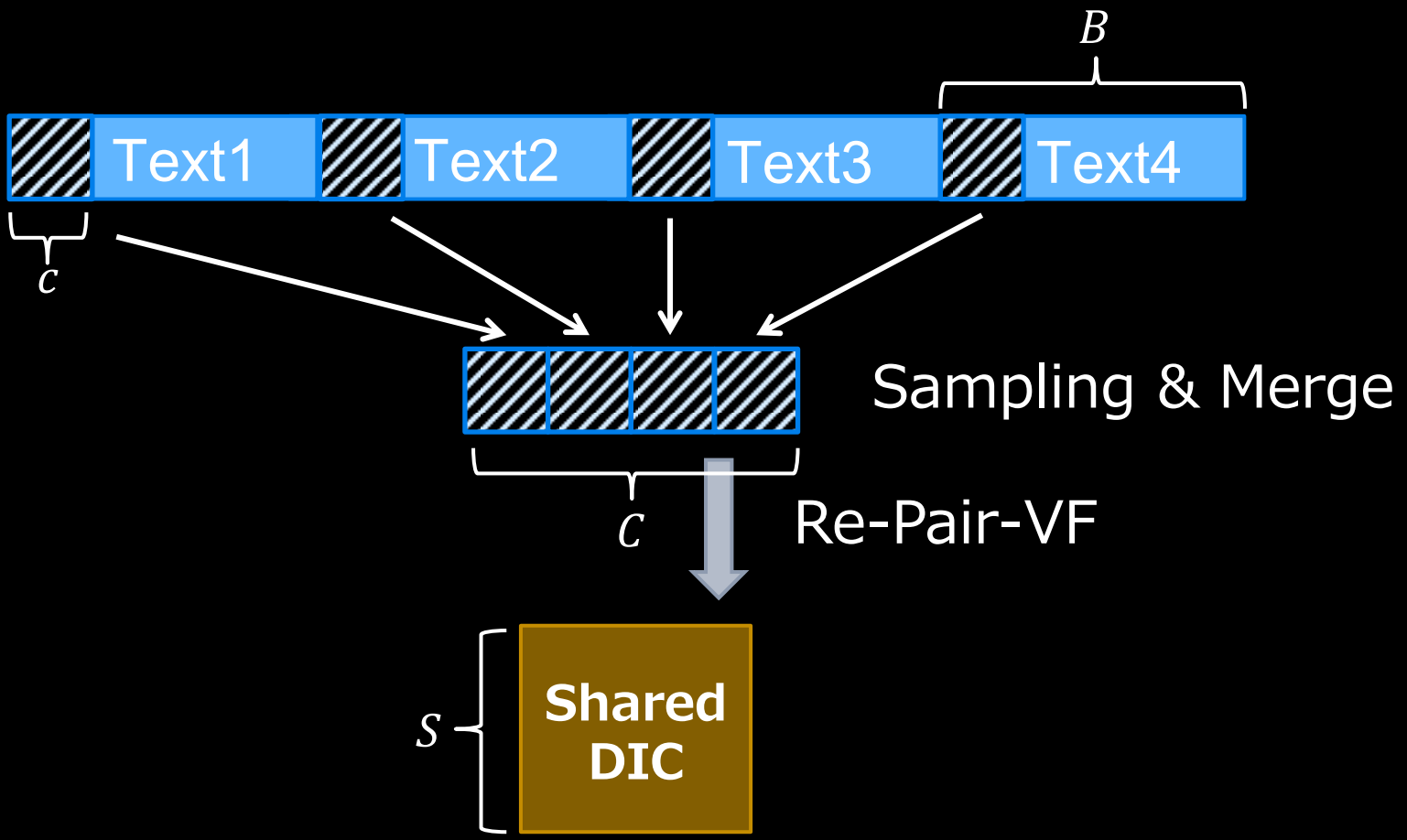
- Blocked-Re-Pair-VF（先頭ブロック法）に対し，共有辞書の構築法を改良し，圧縮パフォーマンスを調査する.

結果

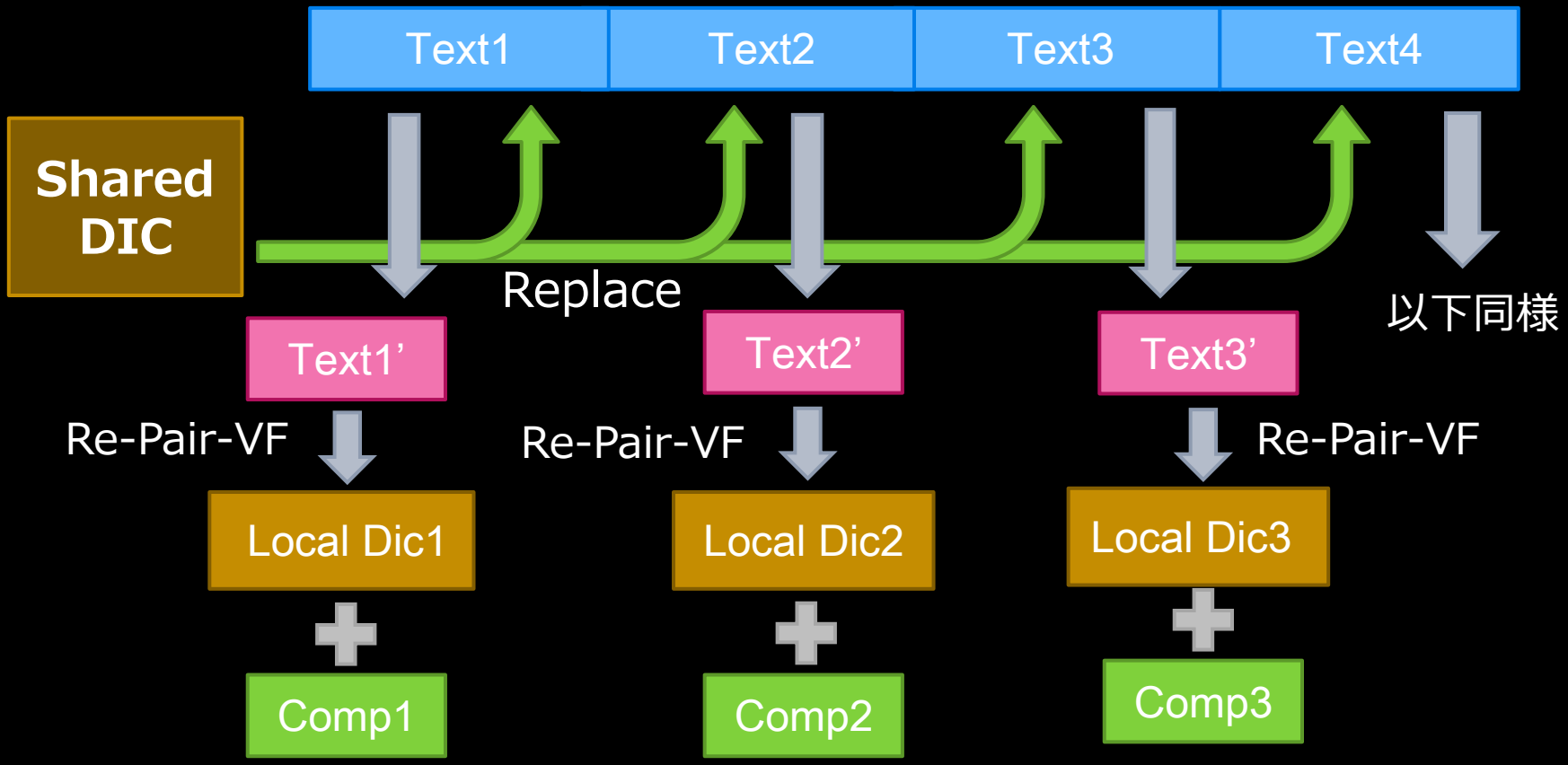
- 改良アルゴリズムサンプル法を考案，計算機実験によりその有用性を示した.
  - 文脈が途中で変わるテキスト，および自然言語テキストに対して圧縮率が最大4%程度改善.

  bzip2に匹敵する圧縮率（約30%）を達成.

サンプル法：共有辞書の作成フェイズ



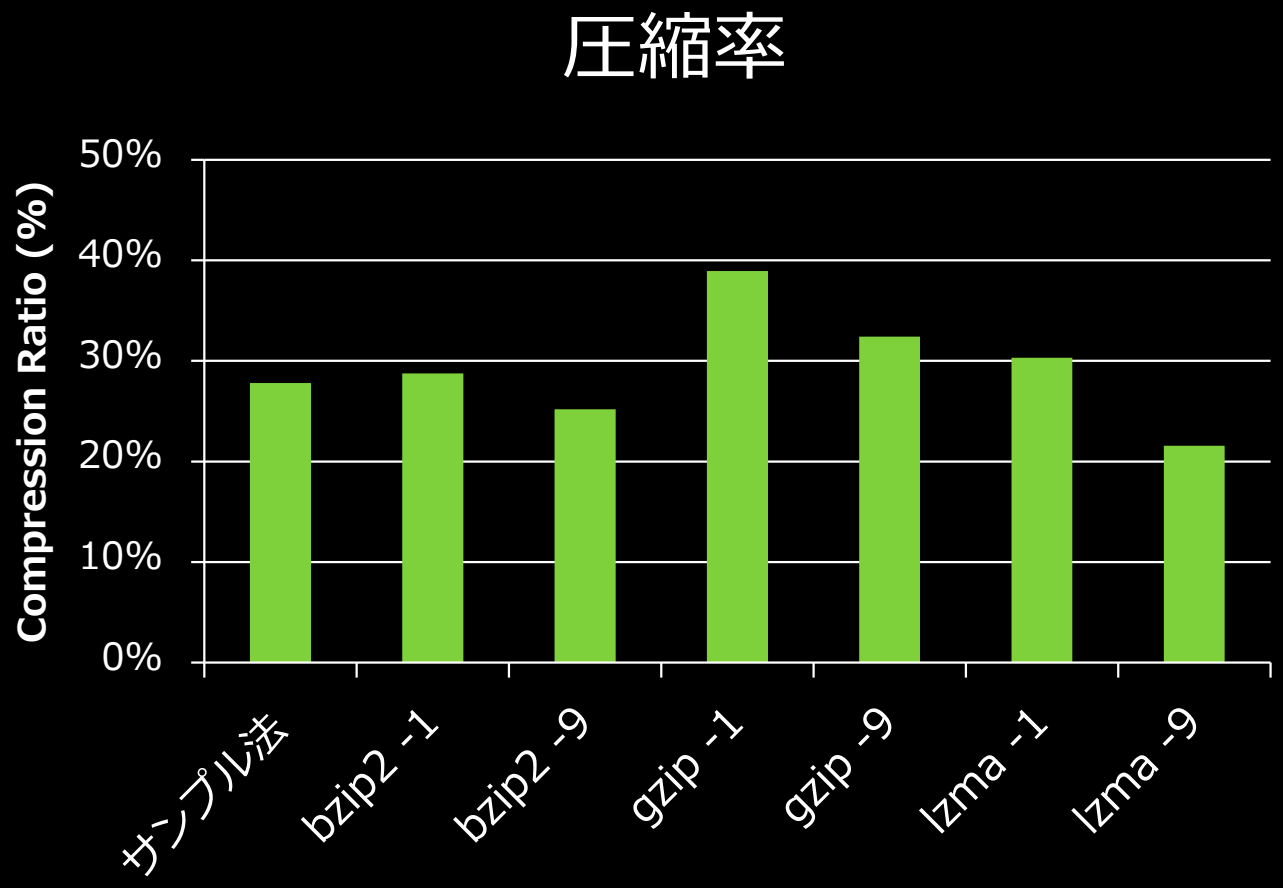
サンプル法：圧縮フェイズ



実験 1

- 目的
  - 既存の圧縮手法とサンプル法の圧縮率を比較する.
- 比較手法
  - gzip
  - bzip2
  - Lzma
- データ
  - Pizza and Chili corpus から取得した，DNAデータ，XMLデータ，および英文の自然言語テキストデータを繋ぎ合わせ，2GBとしたものを用いた(構成は25%，25%，50%).

既存手法との比較



提案手法はbzip2に匹敵する圧縮率（約30%）を達成.

（符号語長19bit，ブロックサイズ128MB，共有辞書の割合3/8，サンプリングテキストサイズ128MB）

実験２

- 目的
  - 変則的なテキストにおけるサンプル法の圧縮パフォーマンス（圧縮率，圧縮時間）を調査する.
- 方法
  - サンプル法および旧手法において，入力パラメータを変化させながら，圧縮時間と圧縮率を計測し比較する.
- データ
  - 実験１と同じ.

変則的なテキストにおける比較

- 圧縮率の比較（サンプリングテキストサイズ128MB）

ブロックサイズ	符号語長	共有辞書の割合	サンプル法	先頭ブロック法
128	18	7/8	29.56%	35.95%
128	19	5/8	28.04%	30.20%
64	19	7/8	29.07%	34.03%
64	22	5/8	30.97%	33.50%
32	20	7/8	29.48%	33.16%

- ほぼ全てのパラメータの組み合わせで圧縮率の改善を確認出来た.

変則的なテキストにおける比較

- 圧縮時間の比較（サンプリングテキストサイズ128MB）

ブロックサイズ	符号語長	共有辞書の割合	サンプル法	先頭ブロック法
128	18	7/8	483.43秒	411.12秒
128	19	5/8	505.33秒	459.41秒
64	19	7/8	493.85秒	428.60秒
64	22	5/8	530.33秒	493.83秒
32	20	7/8	496.20秒	443.74秒

- サンプル法の方が5~10%程度遅い

まとめ

- 結果
  - 大規模テキストに対して，*VF符号による圧縮を行うためのアルゴリズムであるBlocked-Re-Pair-VFを改良した，サンプル法を提案.
  - 共有辞書の改良によって圧縮速度が若干悪化したが，圧縮率は4~20%程度改善された.
  - bzip2並の圧縮率を達成.

- 今後の展望
  - 適切なパラメータの動的な決定法の考案.
  - 共有辞書の作成方法の効率化.

*可変長の文字列に固定長の符号を割り当てる符号化方式