G-TADOC: Enabling Efficient GPU-Based Text Analytics without Decompression

Feng Zhang †, Jidong Zhai ⋄, Xipeng Shen #, Onur Mutlu ⋆, Xiaoyong Du †

†Renmin University of China
⋄Tsinghua University
#North Carolina State University
⋆ETH Zürich

Outline

1. Background
2. Motivation
3. Challenges
4. Our Solution
5. Evaluation
6. Conclusion

1. Background

- TADOC: Text Analytics Directly on Compression

Input:

file0: w1 w2 w3 w1 w2 w4
w1 w2 w3 w1 w2 w4

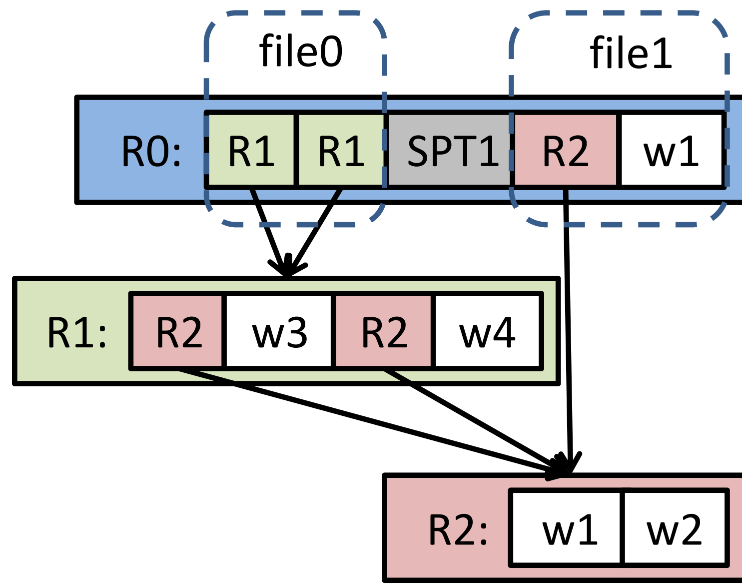
file1: w1 w2 w1

(a) Original data

Rules:

R0 → R1 R1 SPT1 R2 w1
R1 → R2 w3 R2 w4
R2 → w1 w2

(b) TADOC compressed data



(c) DAG Representation

w1: 0 w2: 1 w3: 2
w4: 3 R0: 4 R1: 5
R2: 6 SPT1: 7

(d) Numerical representation

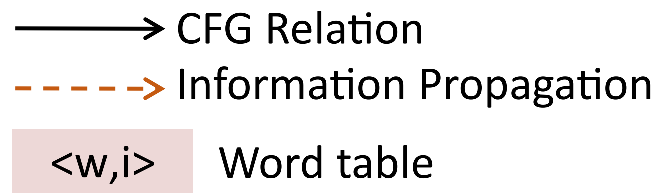
4 → 5 5 7 6 0
5 → 6 2 6 3
6 → 0 1

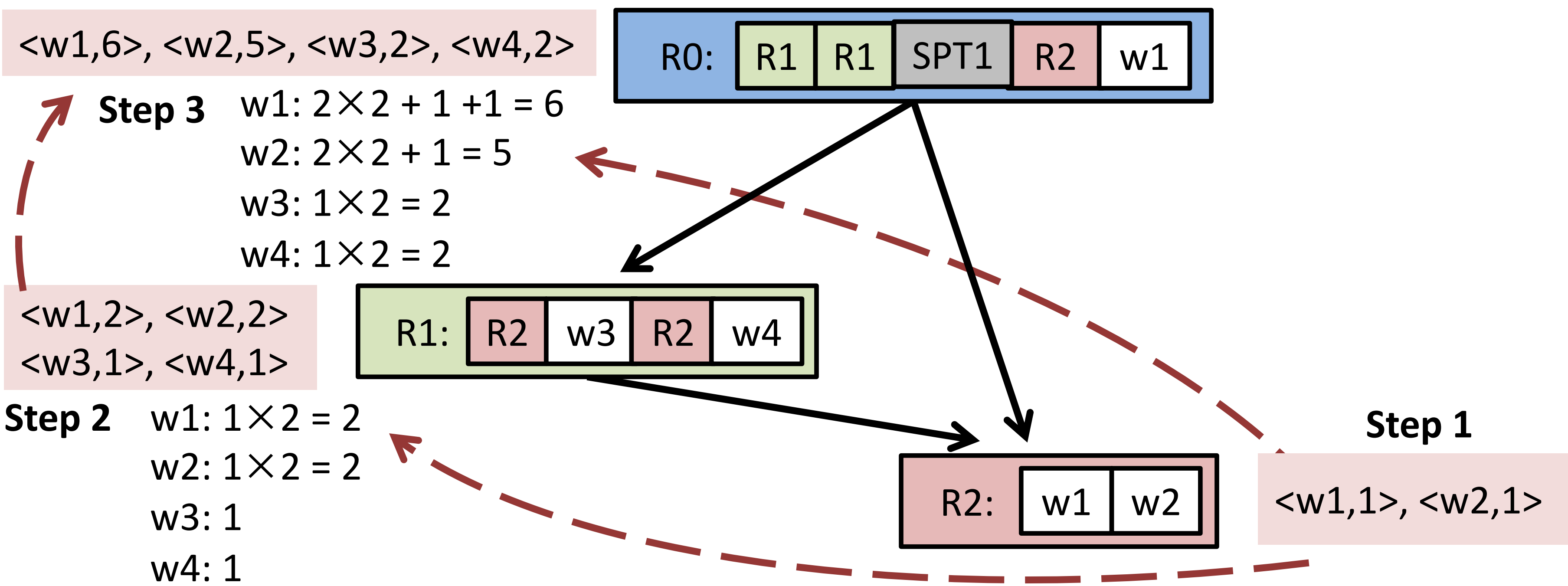
(e) Compressed data in numerical form

[1] Zhang, F., Zhai, J., Shen, X., Mutlu, O., & Chen, W. (2018). Efficient document analytics on compressed data: Method, challenges, algorithms, insights. *Proceedings of the VLDB Endowment*, *11*(11), 1522-1535.

1. Background

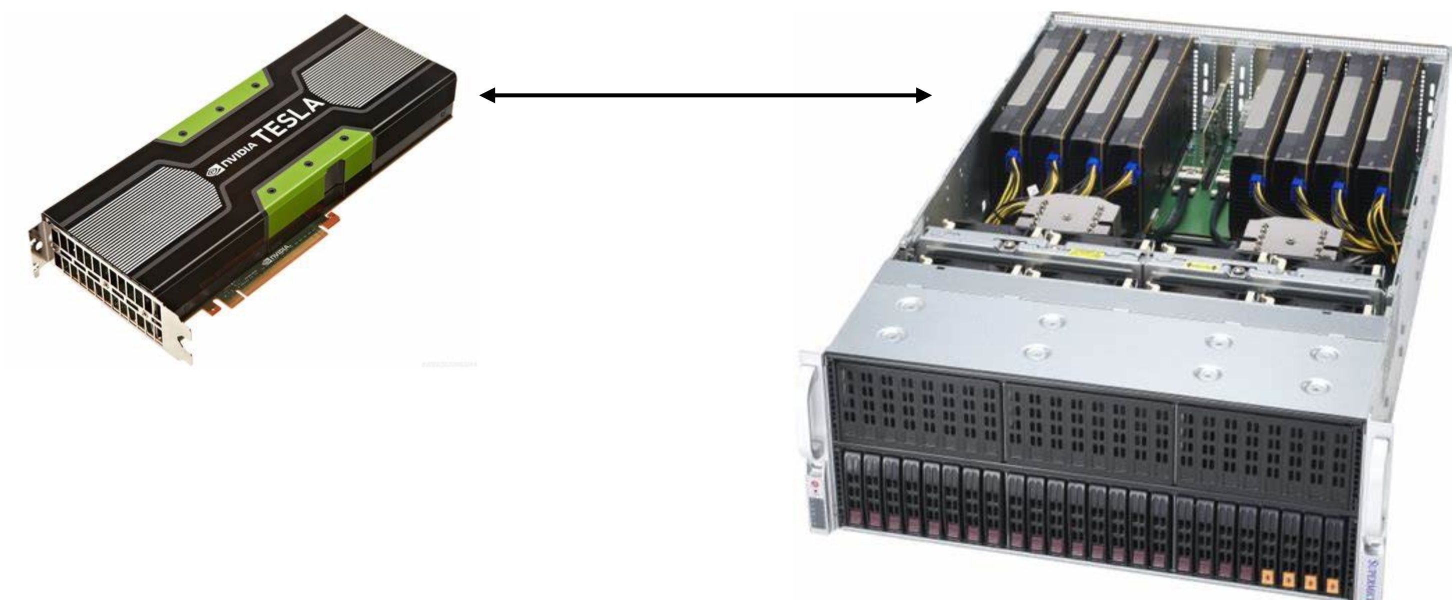
- Example: word count





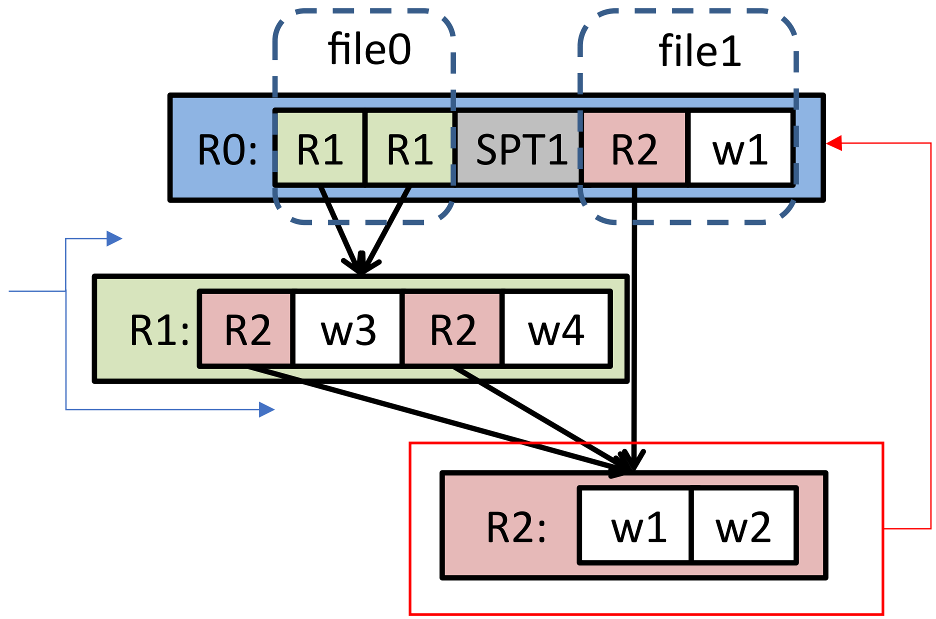
2. Motivation

- GPU – popular in data science
- limited GPU memory
- different from CPUs
- Previous GPU-based methods does not apply



3. Challenges

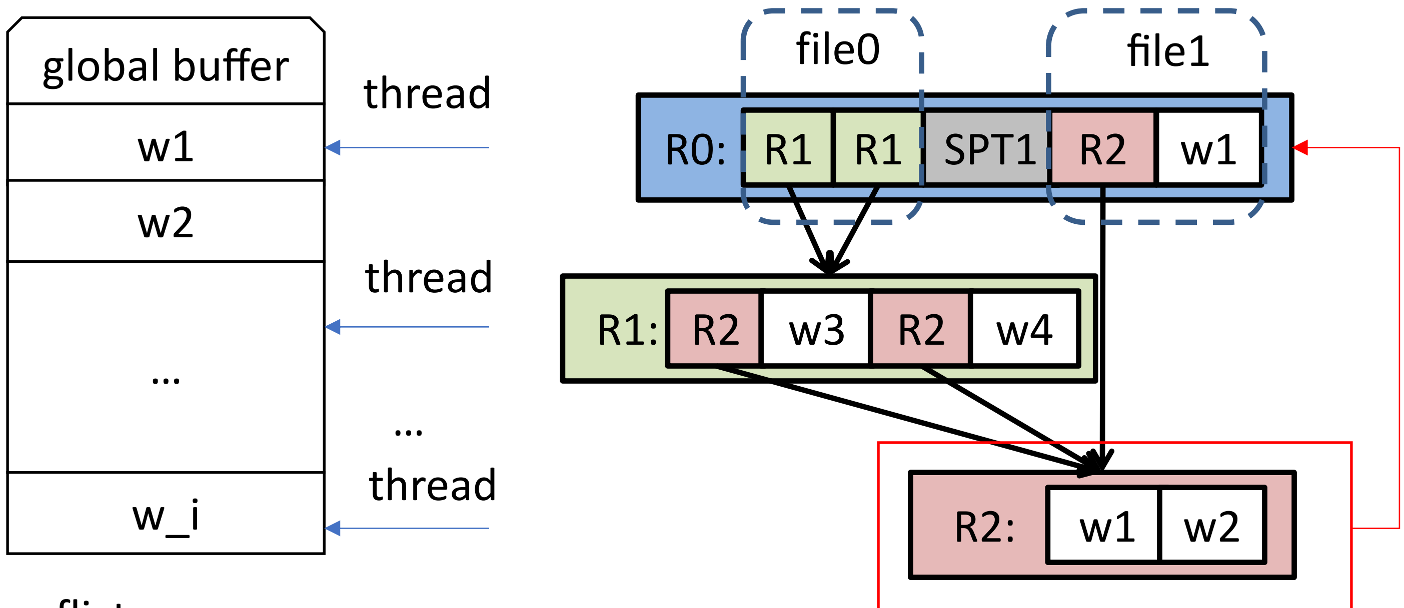
- Challenge 1: GPU parallelism for TADOC
  - Example: R2 depends on R0 and R1, and R1 depends on R0



- dependencies
  - Edges

3. Challenges

- Challenge 2: TADOC final result update conflict of massive GPU threads
  - Example: writing to a global buffer

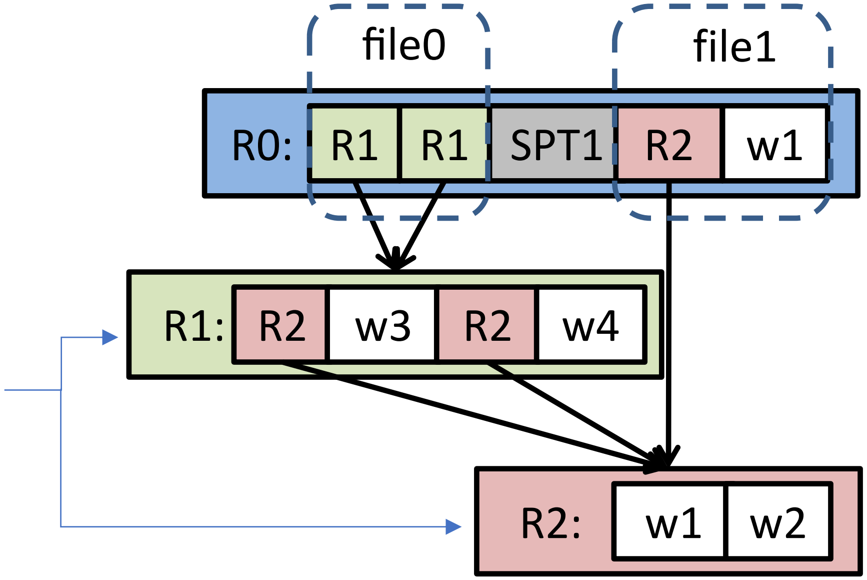


write conflicts

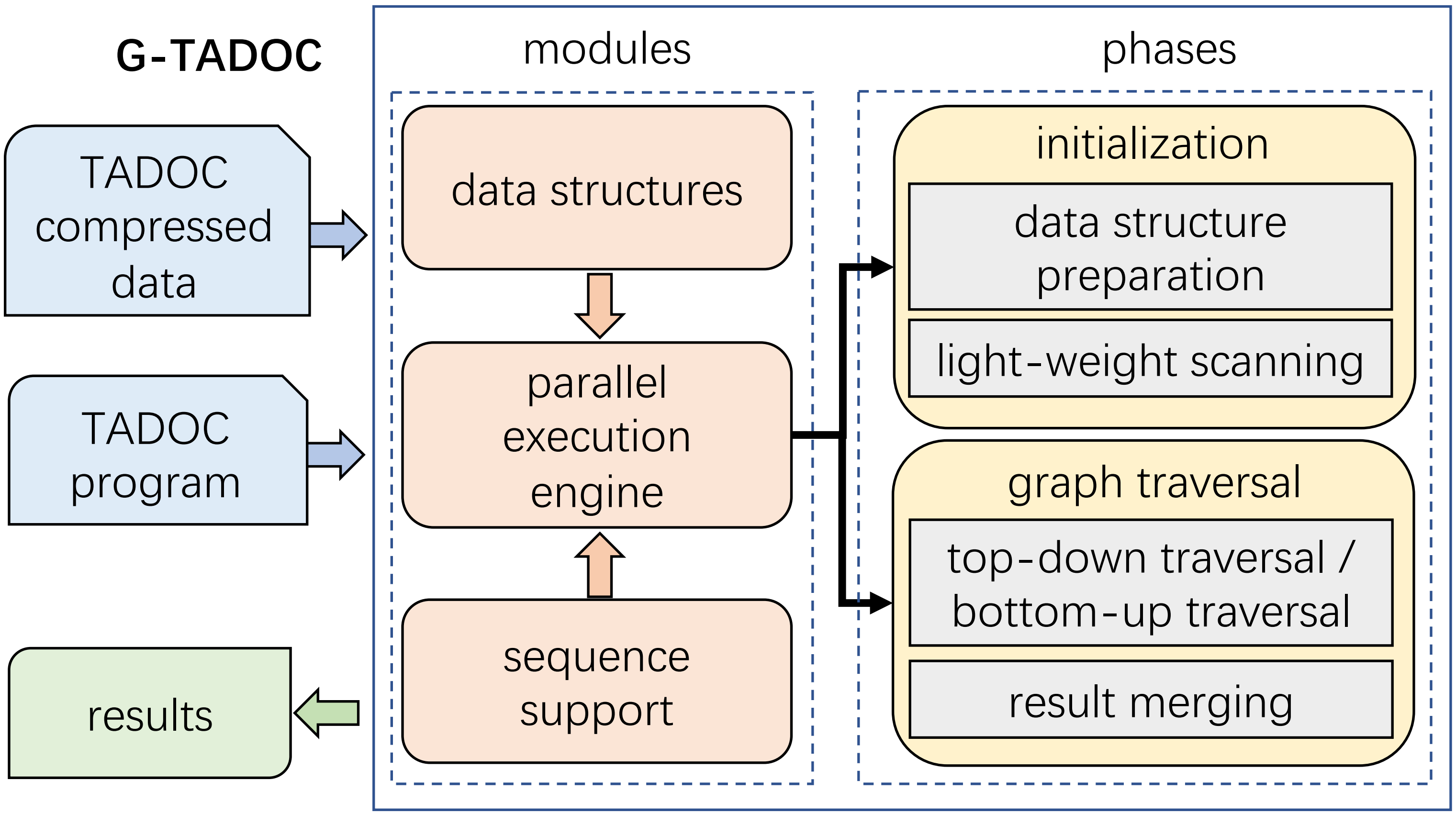
3. Challenges

- Challenge 3: sequence maintenance of TADOC compressed data on GPUs
  - Example: cross-rule sequence

- sequence:
  - w1-w2-w3
    - R1 and R2

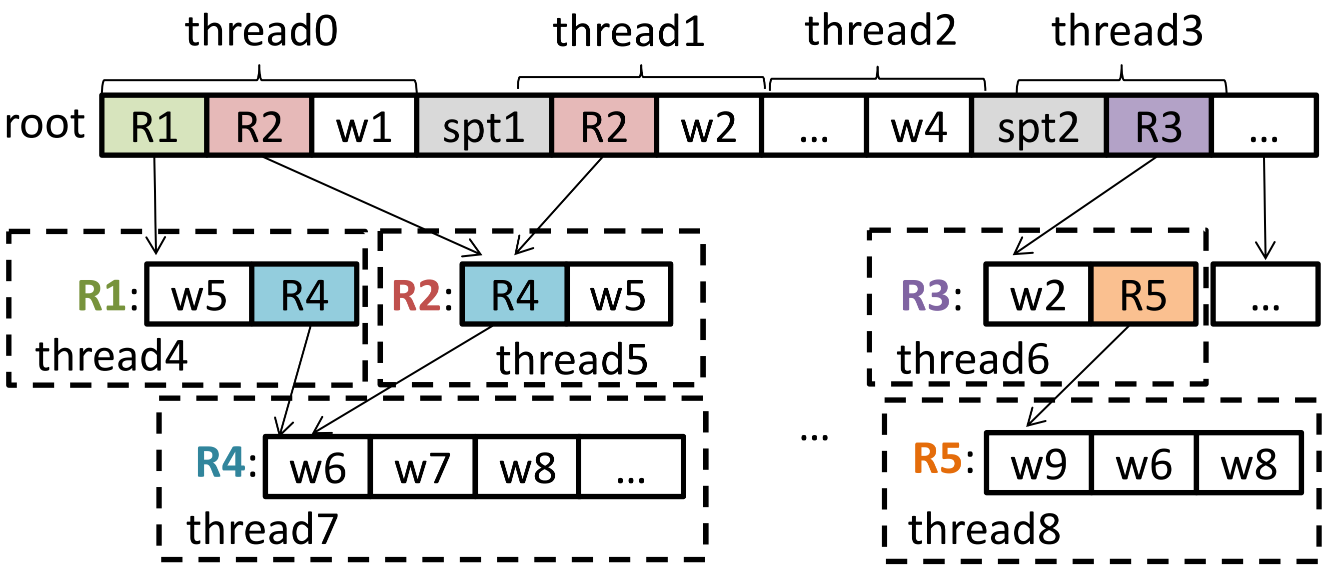


4. Our Solution



4. Our Solution

- Fine-Grained Thread-Level Execution Engine



Fine-grained thread-level partitioning design.

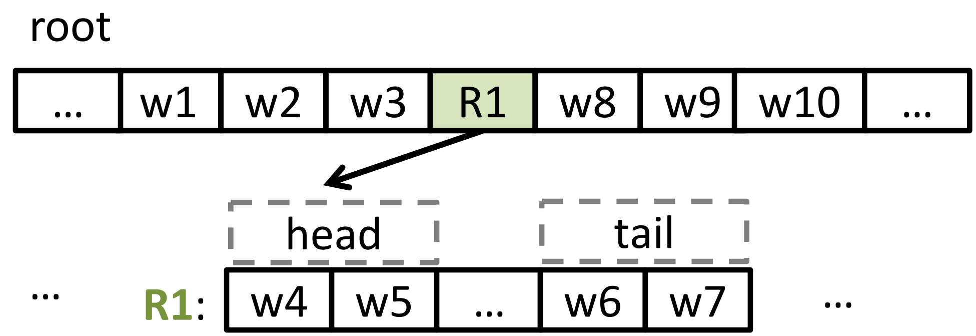
- Top-down / bottom-up traversal design

4. Our Solution

- G-TADOC Data Structures
  - G-TADOC maintained memory pool
  - Thread-safe data structures
  - Head and tail structures for sequence support

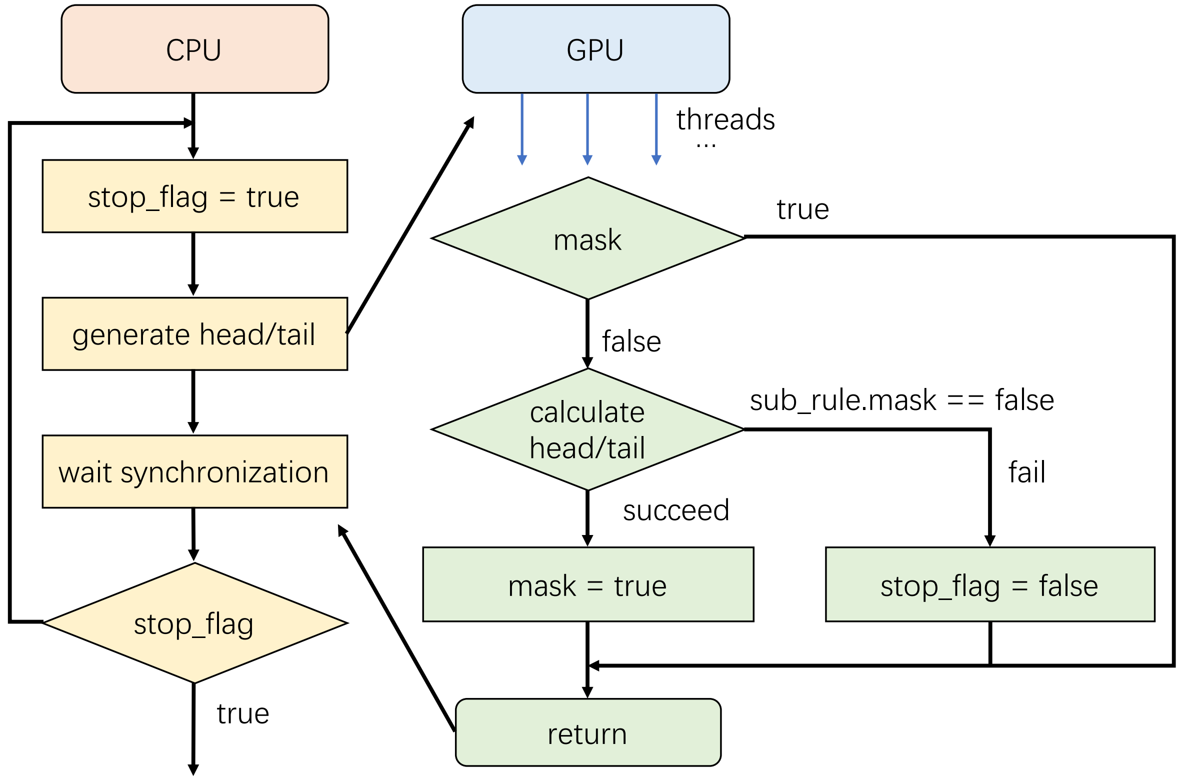
Locks	0	**1**	0	0	0
Entries	-1	0	-1	1	-1
Keys	126	163	**78**		
Values	1	1	**1**	0	0
Next	**2**	-1	**-1**		

(d) Add key = 78 (suppose hash to 1), and value = 1.



4. Our Solution

- Sequence Support in G-TADOC
  - Phase 1: initialization for head and tail buffers
  - Phase 2: graph traversal with sequence support



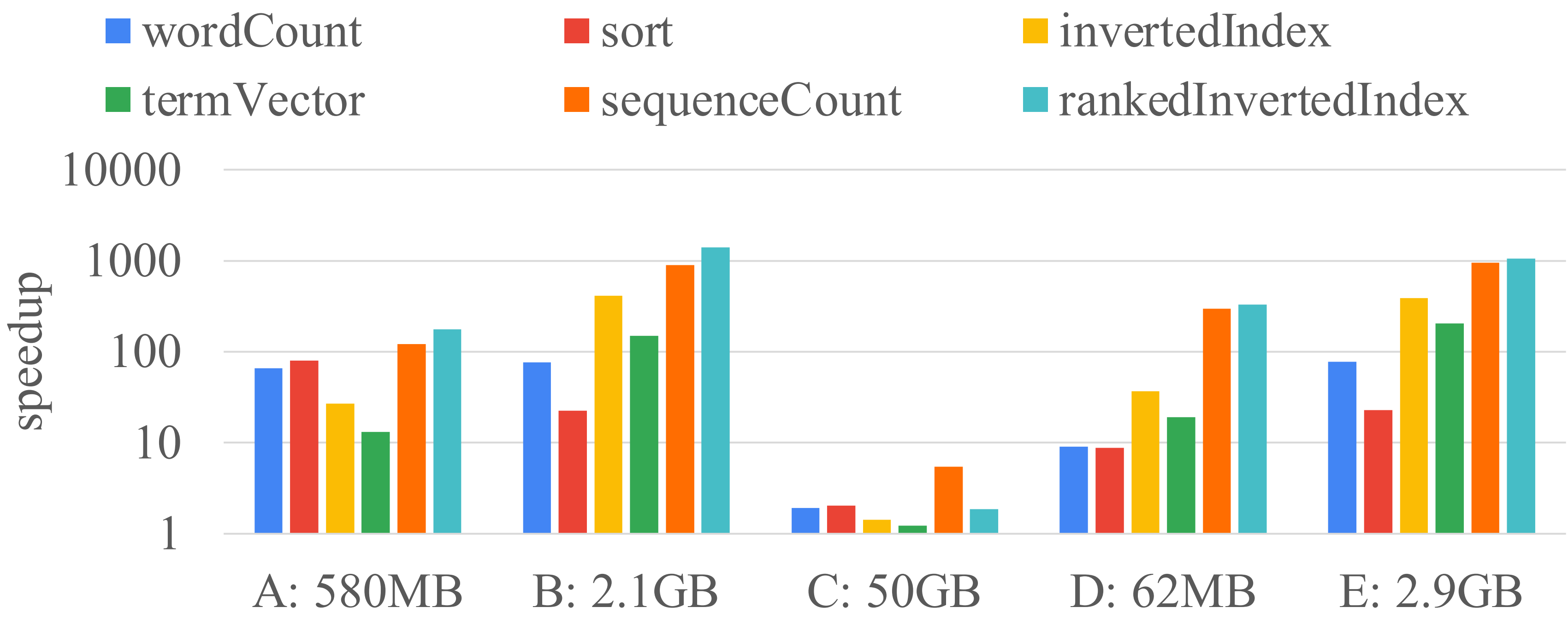
5. Evaluation

- Six benchmarks
  - Word Count, Inverted Index, Sequence Count, Ranked Inverted Index, Sort, Term Vector from [1]
- Five datasets
  - 62 MB ~ 50 GB
- Four platforms
  - three generations of Nvidia GPUs
    - Pascal, Volta, and Turing micro-architectures
  - Spark cluster (10 nodes on Amazon EC2)

[1] Zhang, F., Zhai, J., Shen, X., Mutlu, O., & Chen, W. (2018). Efficient document analytics on compressed data: Method, challenges, algorithms, insights. *Proceedings of the VLDB Endowment*, *11*(11), 1522-1535.

5. Evaluation

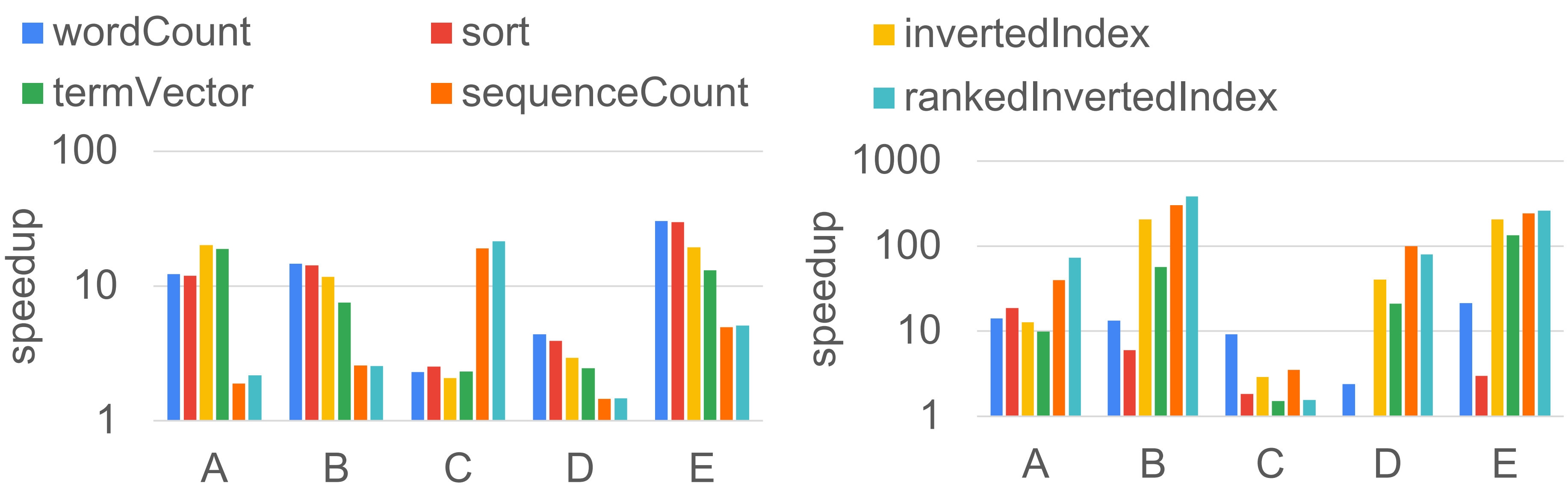
- On average, G-TADOC achieves 31.1× speedup over TADOC.



Turing (GeForce RTX 2080 Ti)

5. Evaluation

- Speedups in different phases

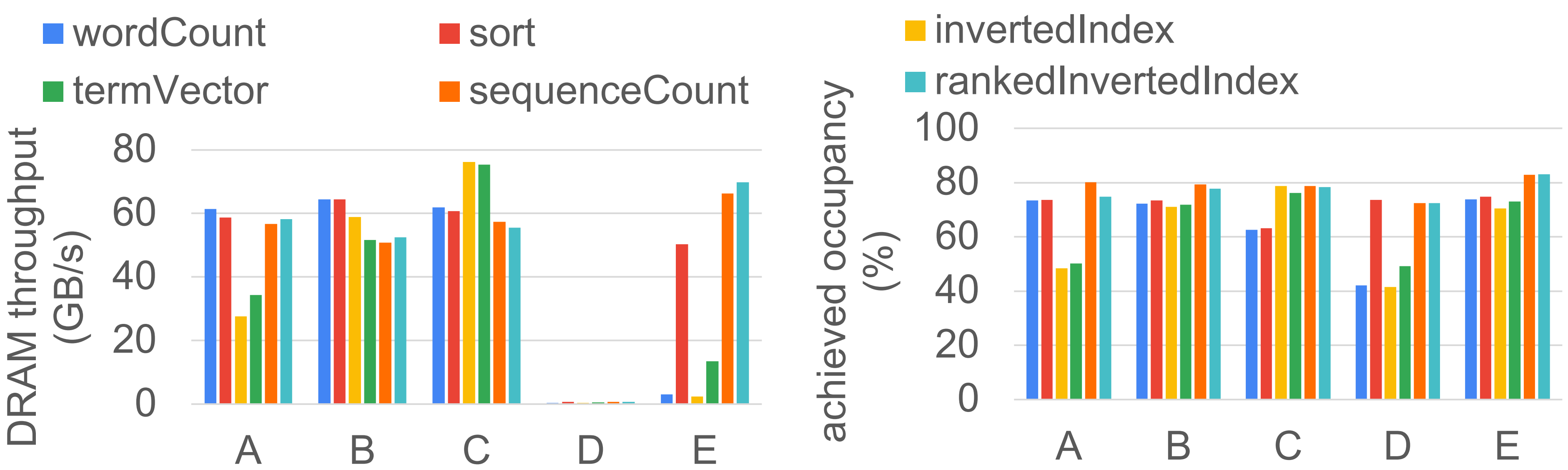


(a) Phase 1: initialization.

(b) Phase 2: traversal.

5. Evaluation

- Analysis of performance metrics



(a) DRAM throughput. (b) Achieved occupancy.

6. Conclusion

- G-TADOC, the first framework enabling efficient GPU-based text analytics directly on compressed data
- Our work can help put much larger content directly in GPU memory.

Thanks!

- Any questions?

Feng Zhang †, Jidong Zhai ⬨, Xipeng Shen #, Onur Mutlu ⋆, Xiaoyong Du †

†Renmin University of China
⬨Tsinghua University
#North Carolina State University
⋆ETH Zürich