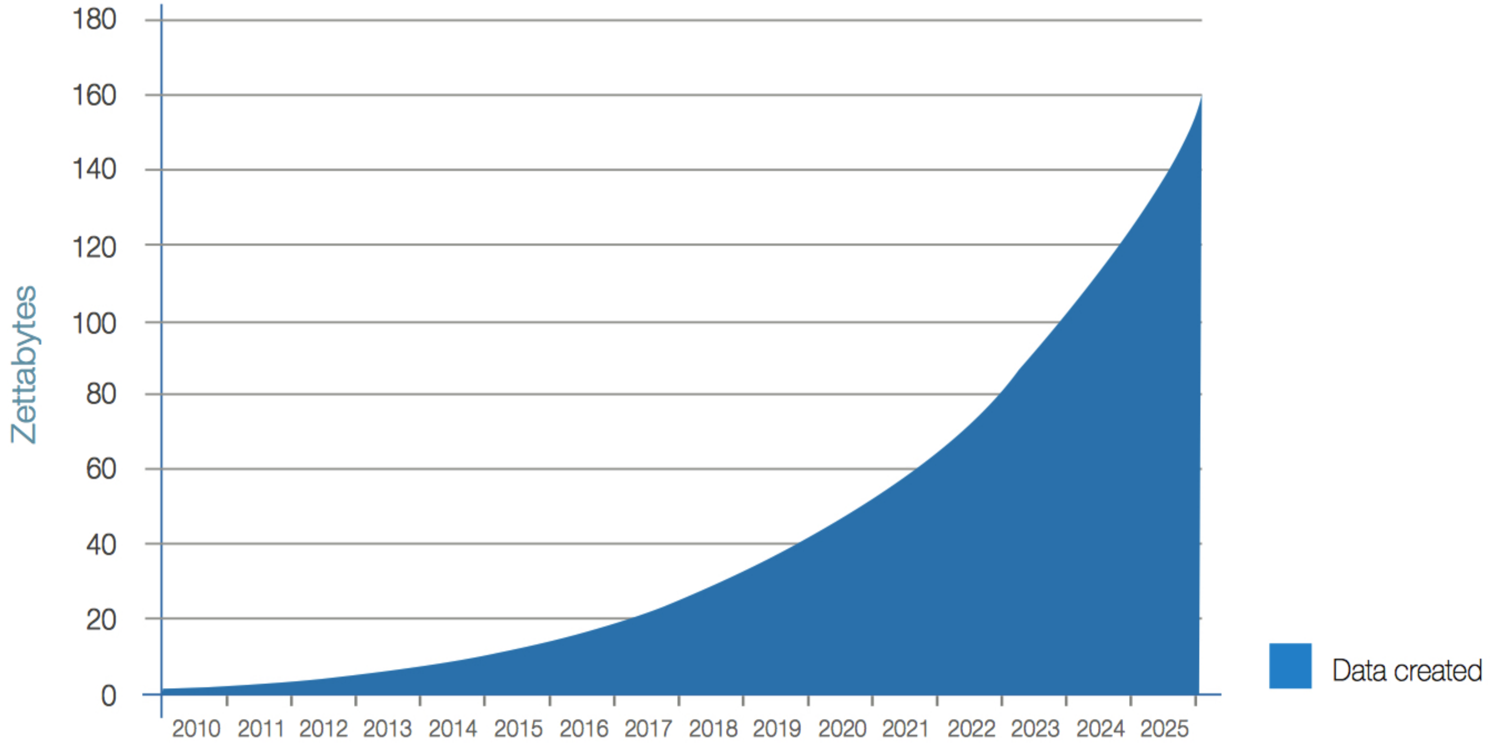
SS-CDC: A Two-stage Parallel Content-Defined Chunking Method for Data Deduplicating

Fan Ni — University of Texas Arlington

Xing Lin — NetApp

Song Jiang — University of Texas Arlington

Data is Growing Rapidly



From storagenewsletter.com

- Most of the data needs to be safely stored.
- **Efficient data storage and management have become a big challenge.**

The Opportunity: Data Duplication is Common

- Sources of duplicate data:
  - The same files are stored by multiple users into the cloud.
  - Continuously updating of files to generate multiple versions.
  - Use of checkpointing and repeated data archiving.

- Significant data duplication has been observed.
  - For backup storage workloads
    - Over 90% are duplicate data.
  - For primary storage workloads
    - About 50% are duplicate data.

The Deduplication Technique can Help

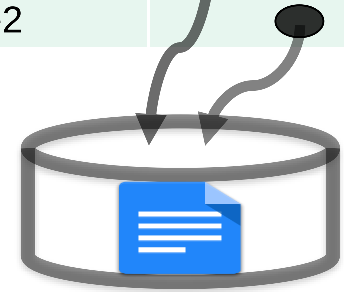
File1

File2

When duplication is detected (using fingerprinting):

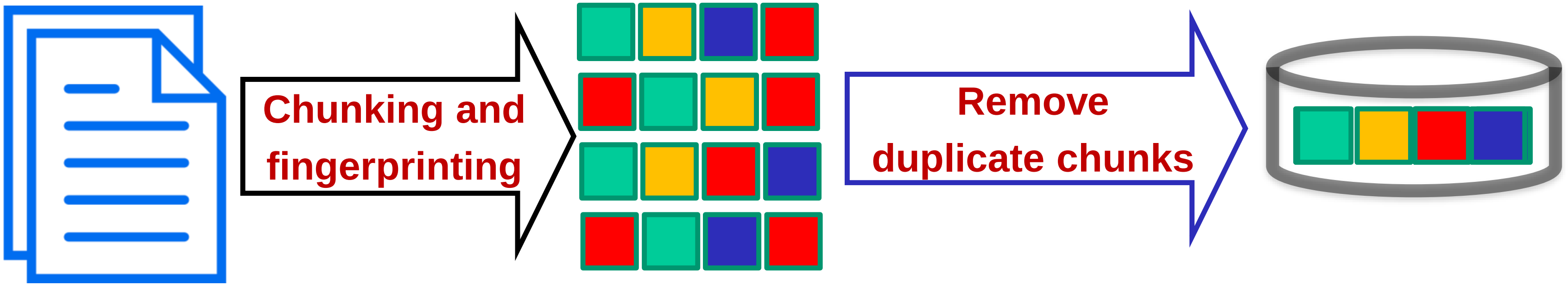
Then only one copy is stored:

Logical	Physical
File1	
File2	



- Benefits
  - Storage space
  - I/O bandwidth
  - Network traffic
- A important feature in commercial storage systems
  - NetApp ONTAP system
  - Dell-EMC Data Domain system
- The data deduplication technique is critical.
  - **How to deduplicate more data?**
  - **How to deduplicate faster?**

Deduplicate at Smaller Chunks …



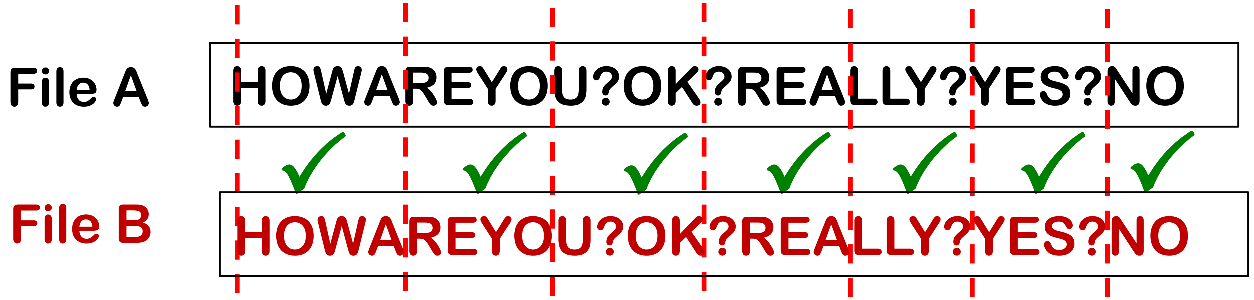
… for higher deduplication ratio

- Two potentially major sources of cost in the deduplication:
  - Chunking
  - Fingerprinting
- **Can chunking be very fast?**

Fixed-Size Chunking (FSC)

- FSC: partition files (or data streams) into equal- and fixed-size chunks.
  - Very fast!

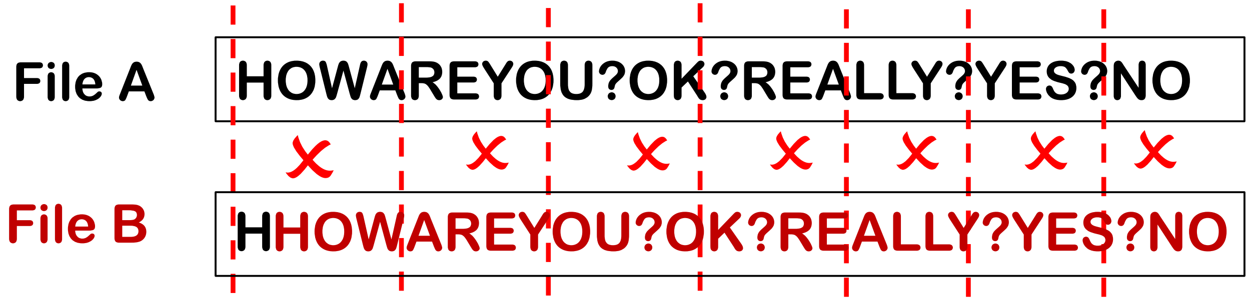
- But the dedup ratio can be significantly compromised.
  - **The boundary-shift problem.**



Fixed-Size Chunking (FSC)

- FSC: partition files (or data streams) into equal- and fixed-size chunks.
  - Very fast!

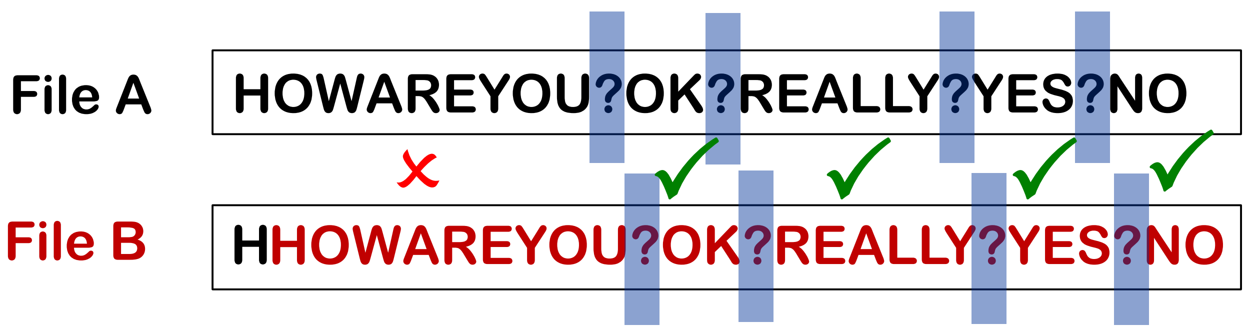
- But the dedup ratio can be significantly compromised.
  - **The boundary-shift problem.**



Content-Defined Chunking (CDC)

- CDC: determines chunk boundaries according to contents (a predefined special marker).
  - Variable chunk size.
  - Addresses boundary-shift problem
  - However, **it can be very expensive**

Assume the special marker is ‘**?**’



Actually the marker is determined by applying a hash function on a window of bytes, such as ***hash(“YOU?”) == pre-defined-value***

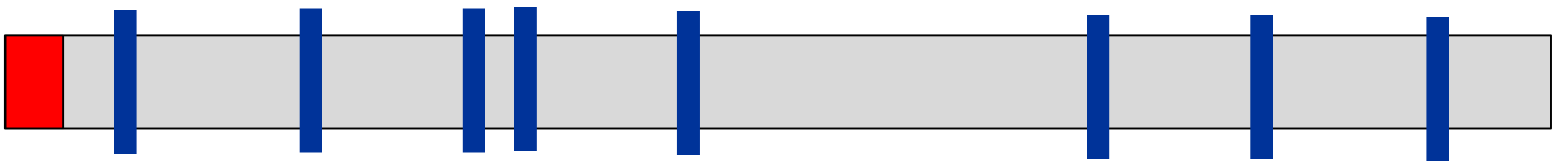
➔ **Even more expensive (likely more than half of the dedup cost!)**

Parallelizing CDC Chunking Operations

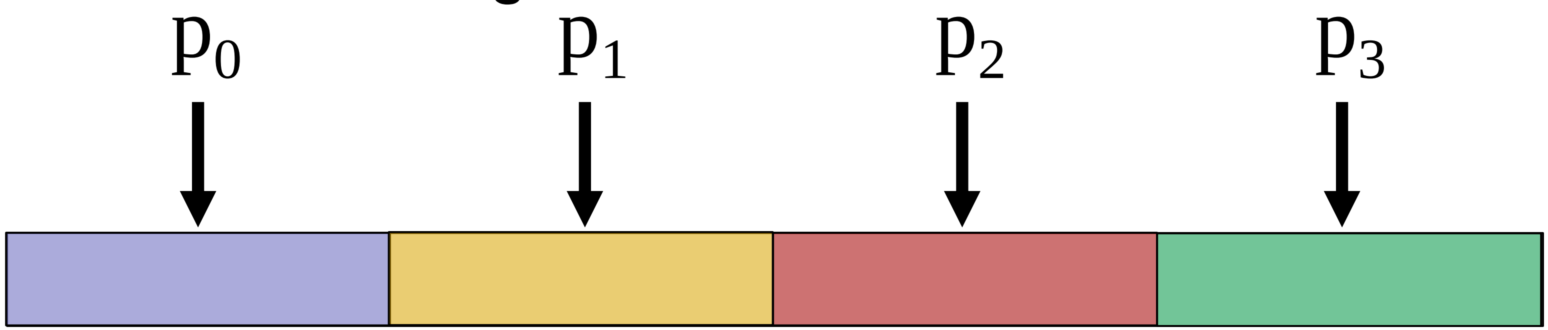
A File

Parallelizing CDC Chunking Operations

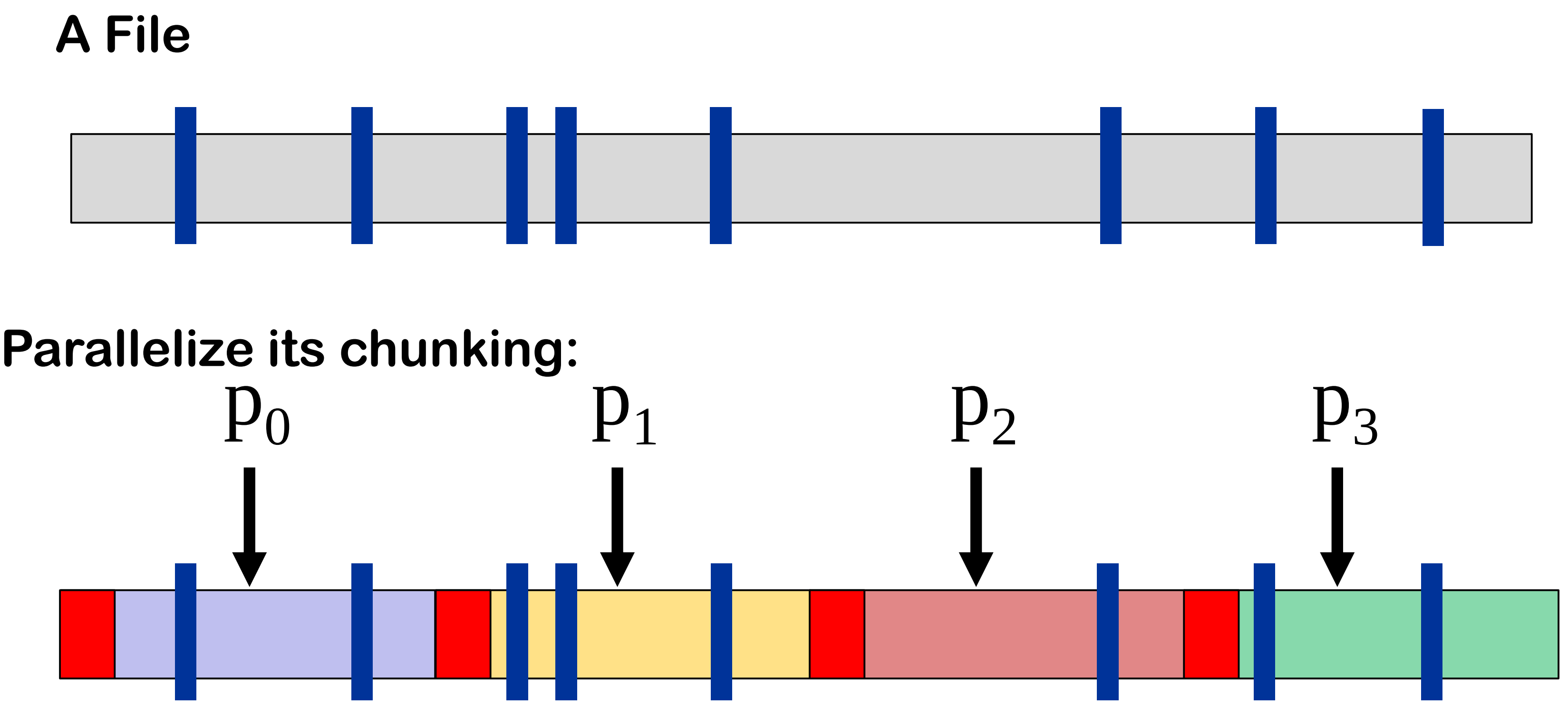
A File



Parallelize its chunking:

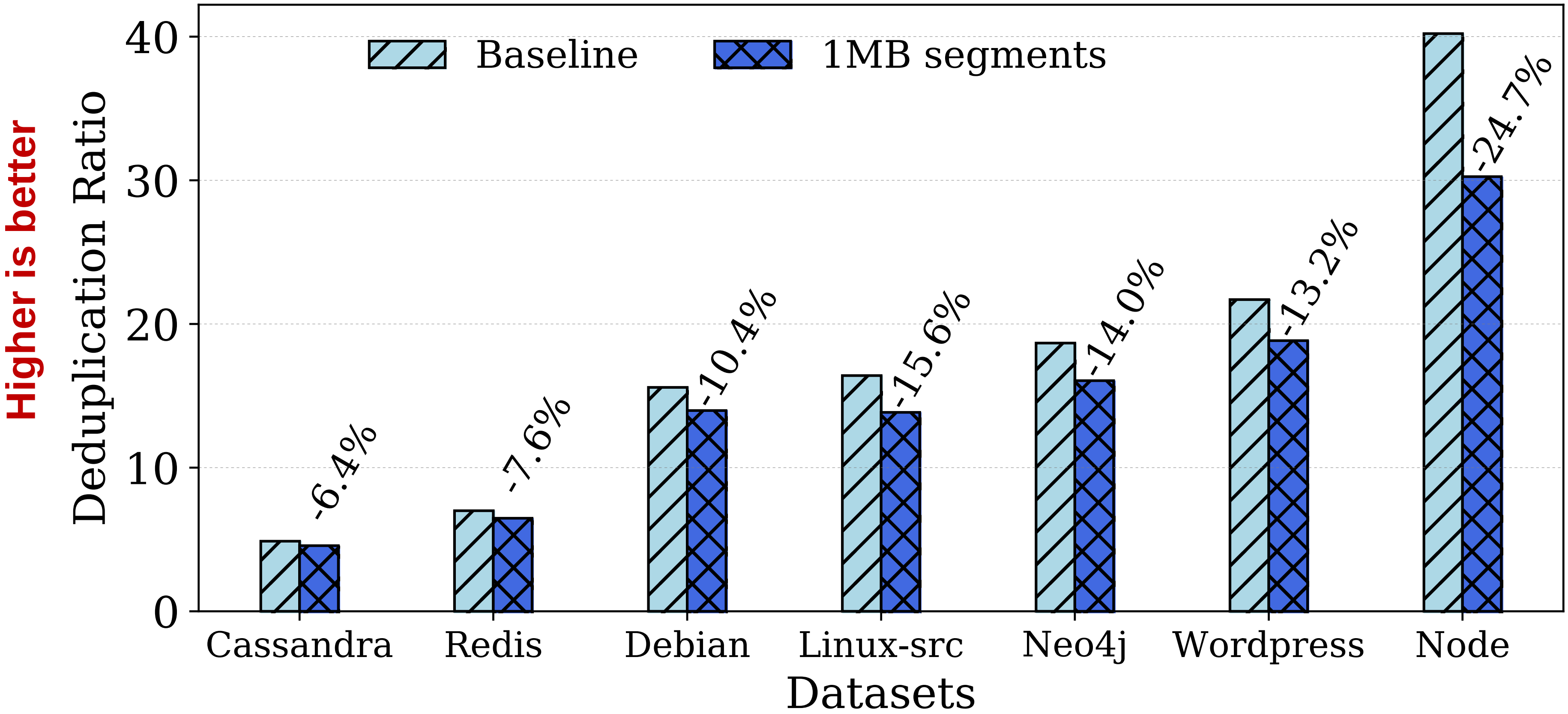


Parallelizing CDC Chunking Operations



However, the parallelized chunking can compromise deduplication ratio.

Compromised Deduplication Ratio

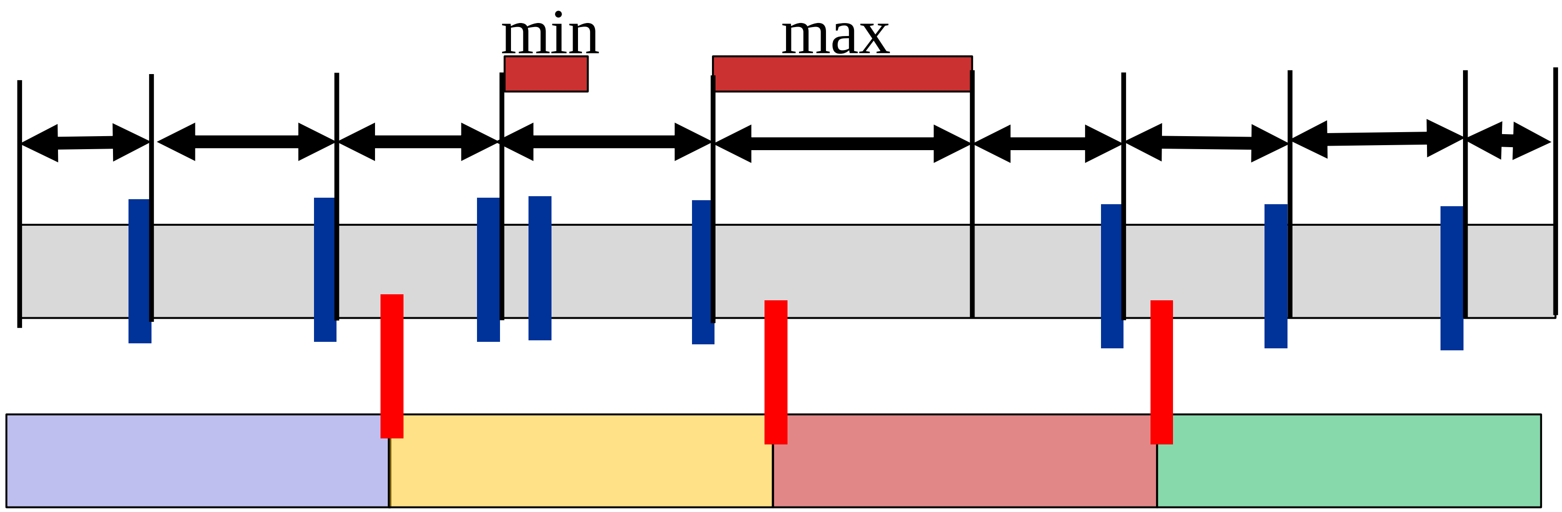


Deduplication ratio = data size before dedup / data size after dedup

Chunks can be Different!

The rule of forming chunks:

– Usually between two adjacent markers.
– But neither too small (≥ Minimum-chunk-size) nor (≤ maximum-chunk-size)
– Inherently a sequential process



The parallel chunking:

– Artificially introduce a set of markers (segment boundaries).
– These maker positions change with data insertion/deletion.
– Partially brings back the boundary shift problem.

The Goal of this Research

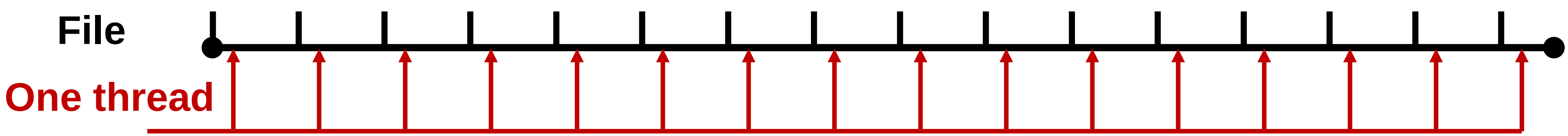
To design a parallel chunking technique that …

- **Does not compromise any deduplication ratio.**
- **Achieves superlinear speedup of chunking operations.**

Approach of the Proposed SS-CDC Chunking

Two-phase chunking:

– **Stage 1: produce all markers in parallel on a segmented file**
  - A thread works on 16 consecutive segments at a time.
  - Use AVX-512 SIMD instructions to process the 16 segments in parallel at a core.



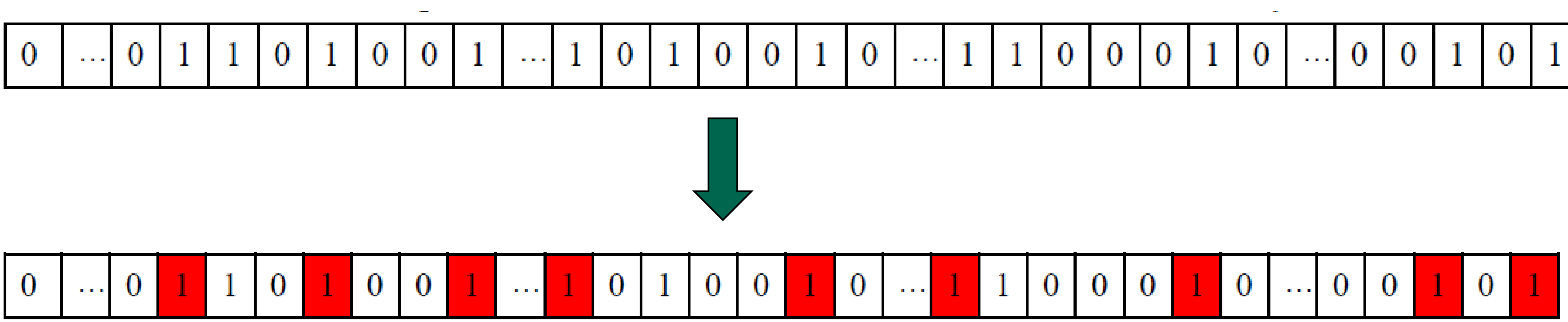
  - The markers are recorded in a bit vector

0	…	0	1	1	0	1	0	0	1	…	1	0	1	0	0	1	0	…	1	1	0	0	0	1	0	…	0	0	1

The Approach of the Proposed SS-CDC Chunking

Two-phase chunking:

- **Stage 2: sequentially determines the chunks based on the marker bit vector**
  - Take account of minimum and maximum chunk sizes



Advantages of SS-CDC

- It doesn't have any loss of deduplication ratio
  – The second stage is sequential.
  – It generates the set of chunks exactly the same the sequential chunking.

- It potentially achieves superlinear speedup.
  – Stage 1 accounts for about 98% of the chunking time.
  – Stage 1 is parallelized across and within cores.
  – With optimization, Stage 2 accounts for less than 2% of the chunking time.

Experiment Setup

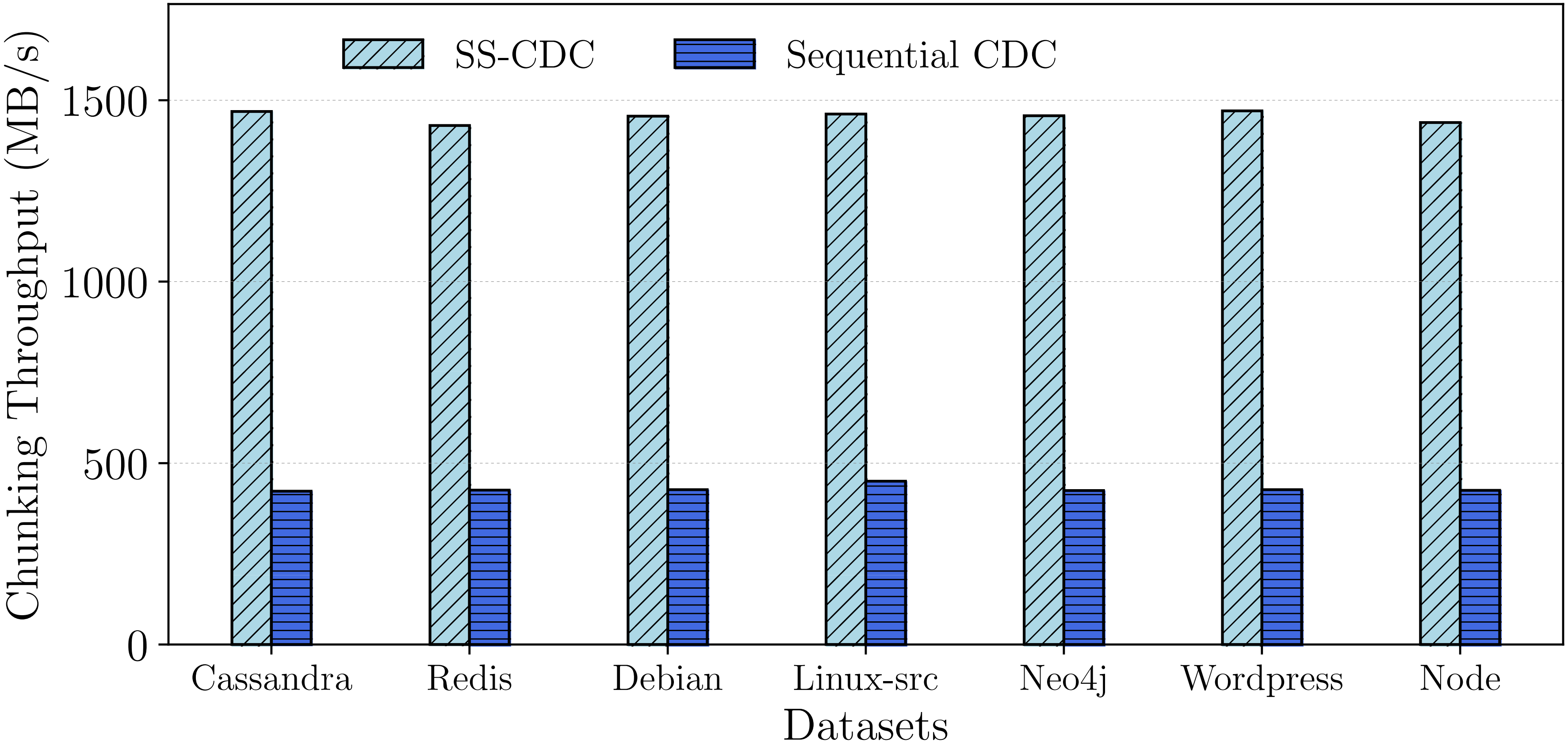
- The hardware
  - Dell-EMC PowerEdge T440 server with 2 Intel Xeon 3.6GHz CPUs
  - Each CPU has 4 cores and 16MB LLC.
  - 256GB DDR4 memory.

- The Software
  - Ubuntu 18.04 OS.
  - The rolling window function is Rabin.
  - Minimum/average/maximum chunk sizes are 2KB/16KB/64KB, respectively.

The Datasets

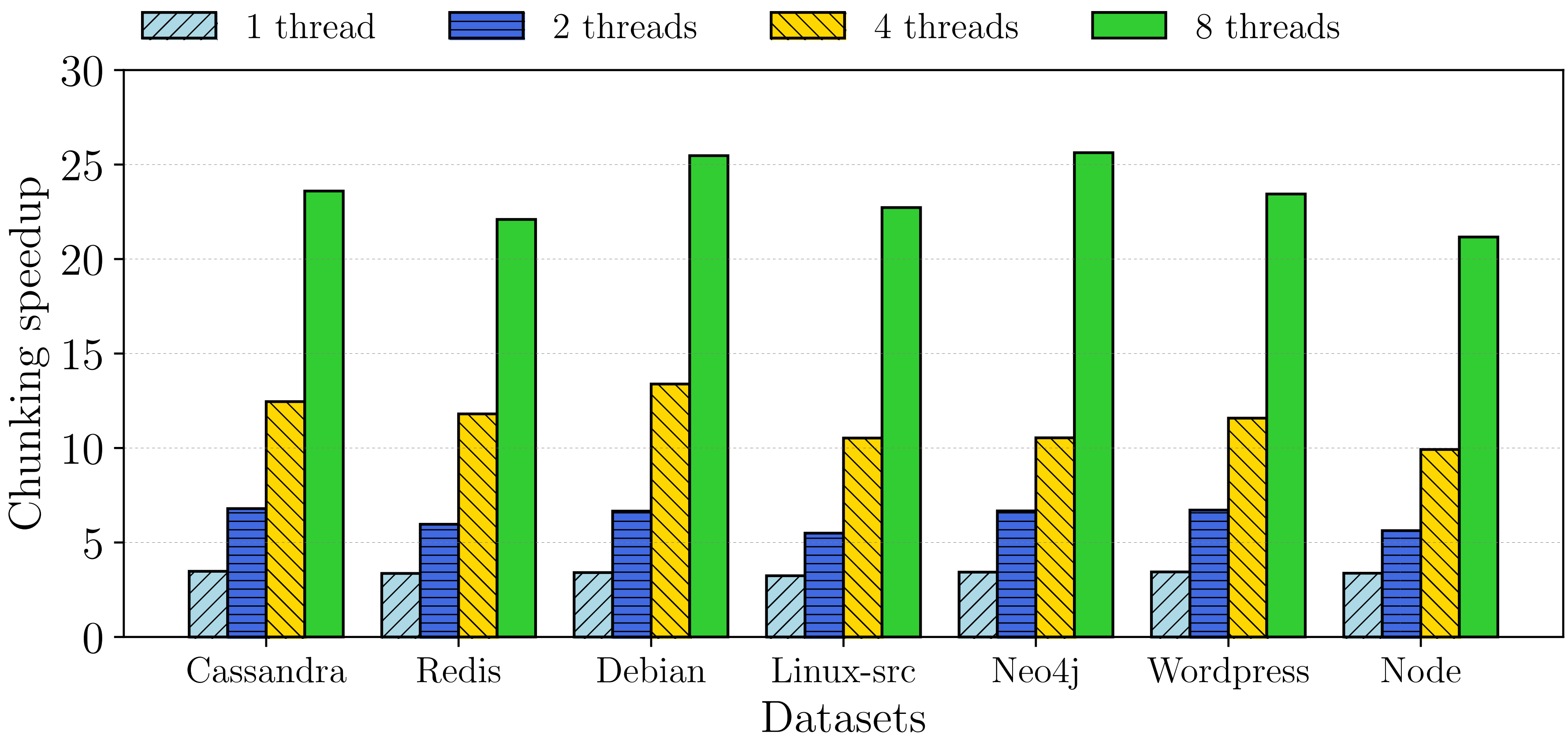
Name	Description
Cassandra	Docker images of Apache Cassandra, an open-source storage system
Redis	Docker images of the Redis key-value store database
Debian	Docker images of Debian Linux distribution (since Ver. 7.11)
Linux-src	Uncompressed Linux source code (v3.0 ~ v4.9) downloaded from the website of Linux Kernel Archives
Neo4j	Docker images of neo4j graph database
Wordpress	Docker images of WordPress rich content management system
Nodejs	Docker images of JavaScript-based runtime environment packages

Single-thread/core Chunking Throughput



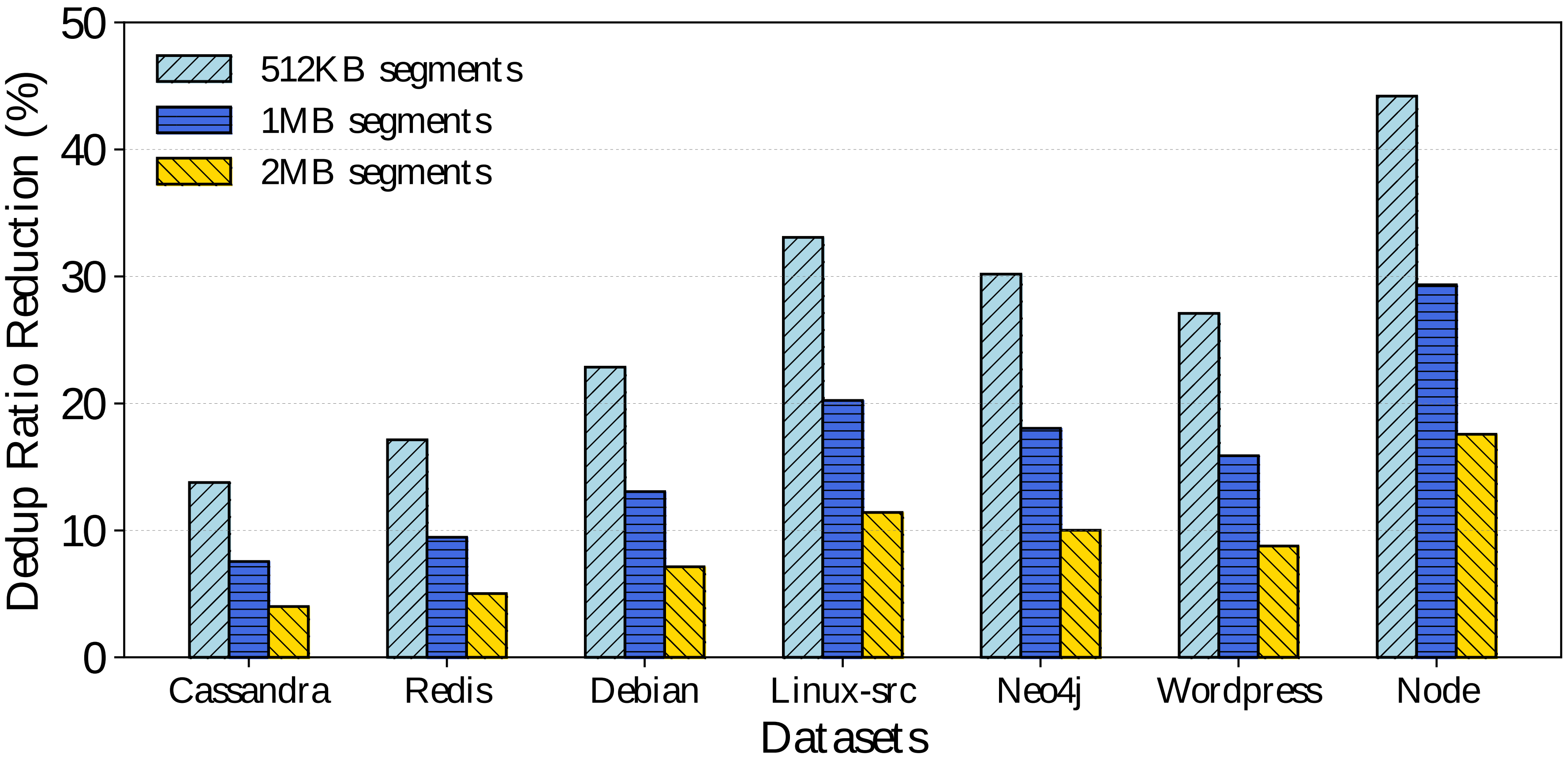
Consistently about 3.3X speedup

Multi-thread/core Chunking Throughput



The chunking speedups are superlinear and scale well.

Existing Parallel CDC Deduplication Ratio Reduction



- Compared to SS-CDC, the reduction can be up to 43%.
- Using smaller segments leads to higher reduction

Conclusions

- SS-CDC is a parallel CDC technique that has
  - **high chunking speed.**
  - **zero deduplication ratio loss.**

- SS-CDC is optimized for the SIMD platforms.
  - Similar two-stage chunking techniques can be applied in other platforms such as GPU.