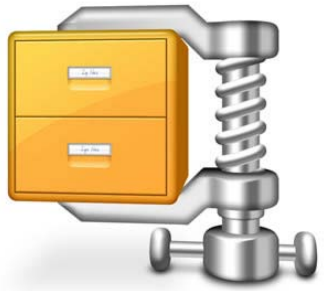
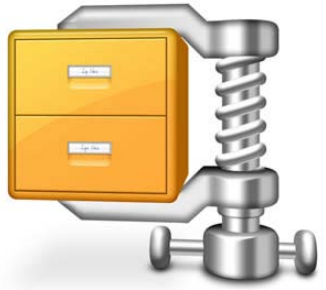
Edelta: A Word-Enlarging Based Fast Delta Compression Approach

Wen Xia, Chunguang Li, Hong Jiang, Dan Feng,
Yu Hua, Leihua Qin, Yucheng Zhang

Outline

- Background and Problems
- Observation and Motivation
- Edelta Design and Implementation
- Performance Evaluation
- Conclusion and Future Work





Dedup vs. Delta Compression

- In recent years, dedupication and delta compression are gaining increasing attentions

	Delta Compression	Data Deduplication
Target	Similar data	Duplicate data
Processing Granularity	String	Chunk/File
Representative Methods	KMP based Copy/Insert	CDC & Secure Fingerprint
Scalability	Weak	Strong
Representative Prototypes	Xdelta, Zdelta	LBFS, DDFS

➢**Data deduplication runs much faster than delta compression**

➢**Delta compression is able to eliminate more redundancy among non-duplicate but similar chunks (about 2-3X more)**

Can delta compression run faster than deduplication?

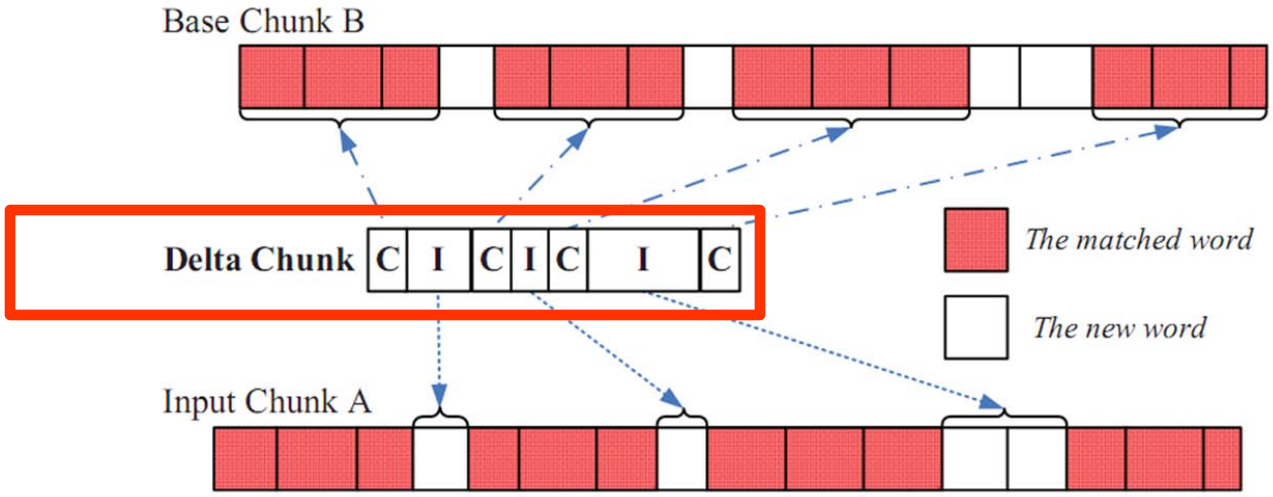
State of the Art on Delta Compression

$$A_i + A_b \xrightarrow{\text{Delta}} \Delta_{b,i} \qquad \Delta_{b,i} + A_b \xrightarrow{\text{Reverse delta}} A_i$$

- Delta encoding
  - Xdelta, Zdelta, Ddelta(Performance'14)
- Cache compression
  - Difference Engine (OSDI'08)
  - I-CASH (HPCA'12)
- WAN optimization/backup storage
  - Dropbox…
  - SIDC (FAST'12, HotStorage'12)

Delta Encoding

- Our Previous Work: Ddelta
  - ➢**Use Gear-based CDC to fast partition strings (words)**
  - ➢**Encode the Matched /New words into Copy/Insert messages**



About 3X faster than Xdelta, Cloud it be more faster ??

Observation and Motivation

- **Observation I: In Ddelta, 96% of the time overhead is from Chunking (~45%), hashing (~16%), and indexing (~35%)**
- **Observation 2: "Copy "is very long while "Insert" is short,**

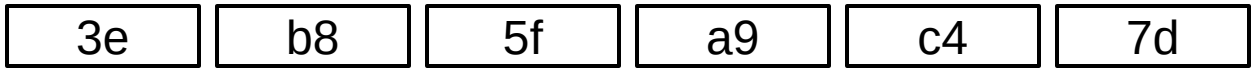
Average length of the grouped C/I messages

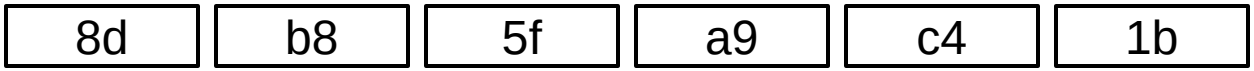
Dataset	LX	SC	GC	EC	GD
Copy	10K	5K	3k	18K	10K
Insert	123	340	133	124	173

➢**Motivation: Can we exploit word-content locality to reduce some unnecessary computation operations.**

An Example

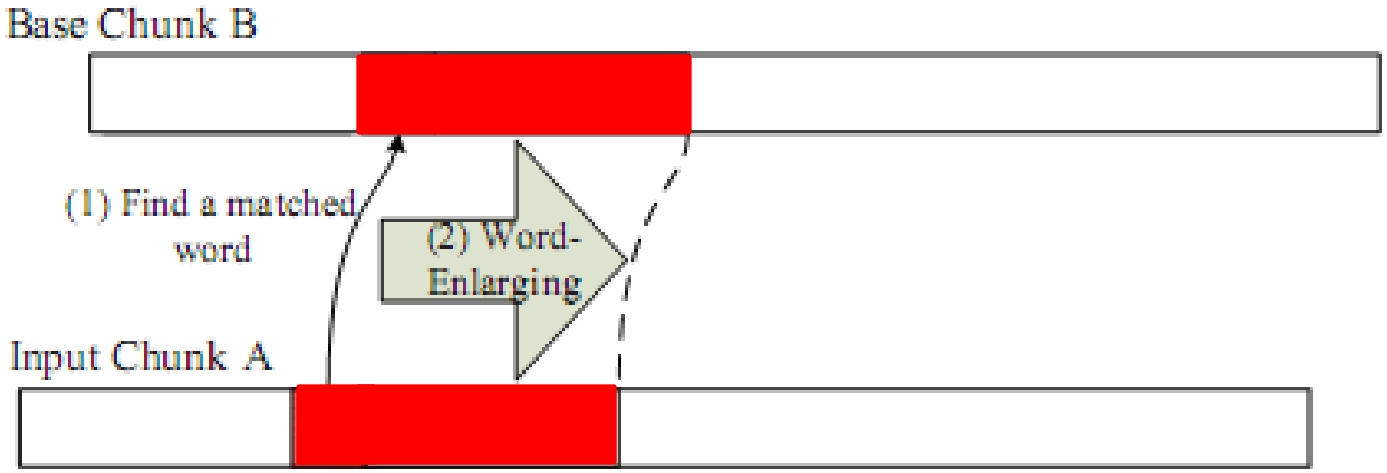
- For those contiguous duplicate words {b8, 5f, a9, c4}, the chunking, hashing, and indexing for the words {5f, a9, c4} would be unnecessary by directly enlarging the detected word {b8}, which is just a fast byte-wise comparison.





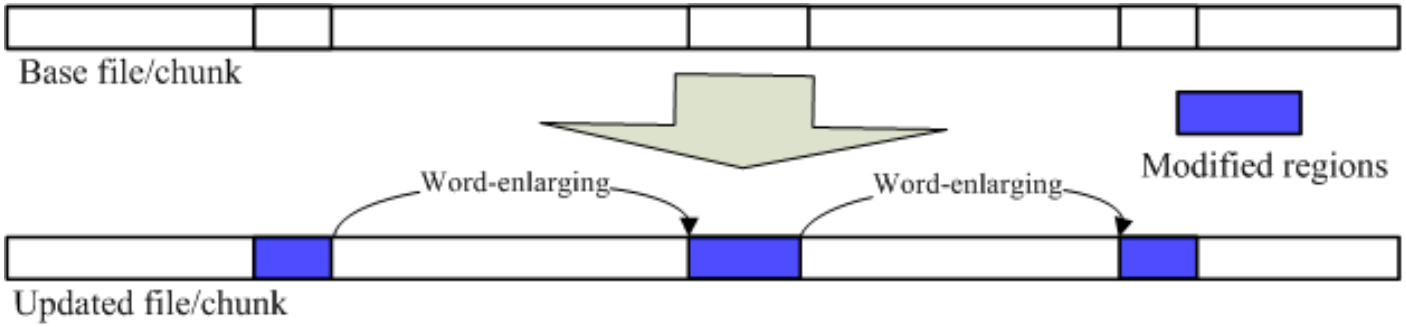
Implementation of Edelta

- **We implement Edelta on top of our previous work Ddelta**
- **For the two known or detected similar chunks, Edelta consists of two key steps: Find a matched word and then enlarge the word**



Continue..

- Step (1): Tentatively detects a duplicate word by Ddelta's scheme.
- Step (2): Directly enlarge the detected word into a much longer one and thus avoid the word-matching operations in the enlarged regions.



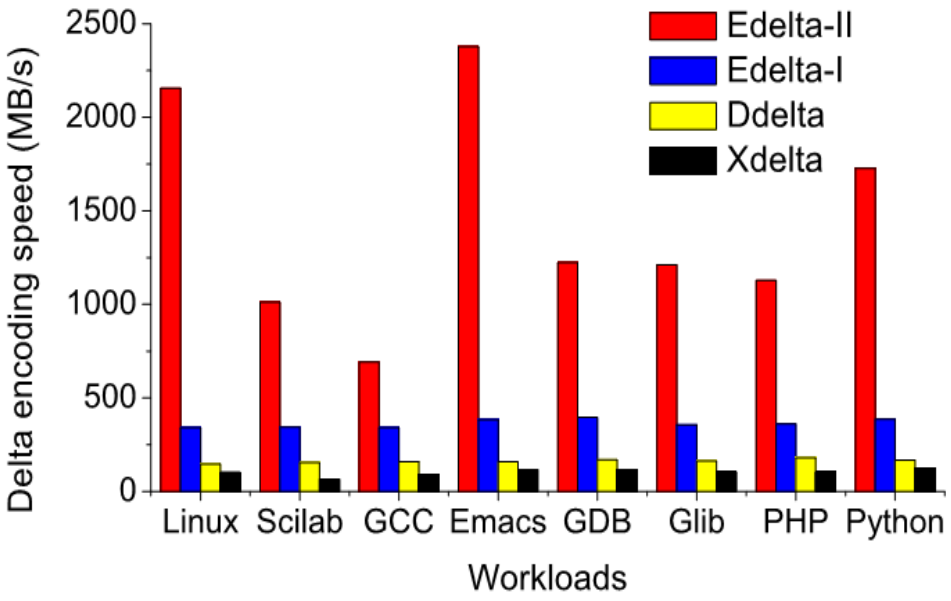
Therefore, Edelta is able to quickly identify the modified areas for delta compression by word-enlarging.

- Scheme I only word-enlarges the input data file/chunk
- Scheme II word-enlarges both the input and base files/chunks

Evaluation

- Metrics: Compression ratio and encoding speed
- Experimental Setup
  - Intel i7 processor, 16GB RAM, two 1TB 7200rpm hard disks, and a 120GB SSD of Kingston VP200S37A120G.
- Two case studies
  - 1. Delta compressing the updated tarred files
    - Datasets: linux, GDB, GCC, etc. tarred files
  - 2. Delta compressing the non-duplicate but similar Chunks
    - Datasets: RDB, VM images, Linux
    - First deduplication, and then resemblance detection, delta encoding the detected chunks.

Case Study I



Dataset	Xdelta	Edelta-II
Linux	99.81%	98.72%
SciLab	97.08%	95.05%
GCC	99.69%	97.04%
Emacs	99.89%	99.32%
GDB	99.87%	98.91%
GLib	99.74%	98.08%
PHD	99.62%	97.75%
Python	99.85%	99.03%

Compression ratio

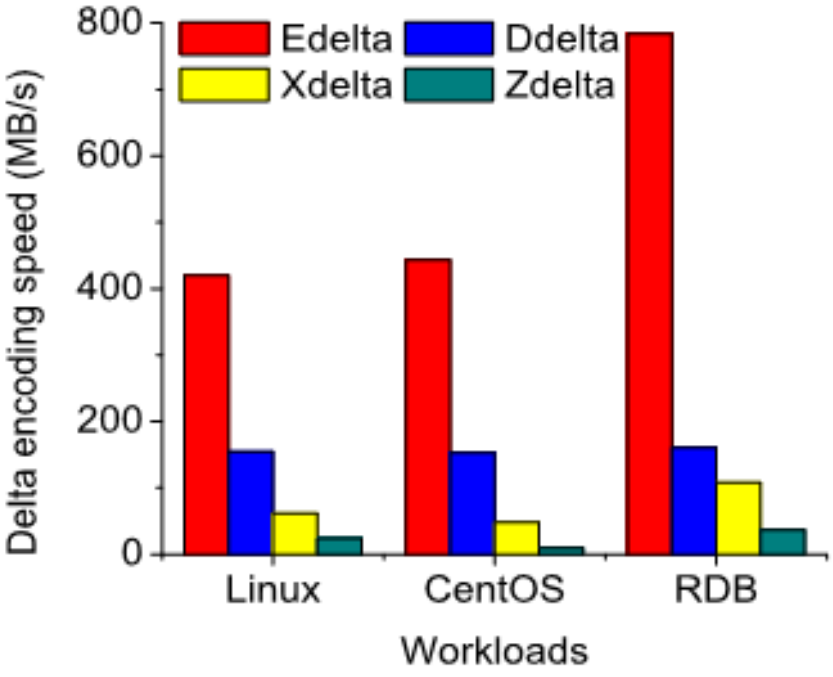
- 5-20X Improvement

Dataset	LX	SC	GC	EC	GD
Copy	10K	5K	3k	18K	10K
Insert	123	340	133	124	173

- Only 1-2% decrease

Case Study II

- Post-deduplication delta compression
  - Dedup factors of the three datasets are 44.7, 2.0, and 22.4 respectively

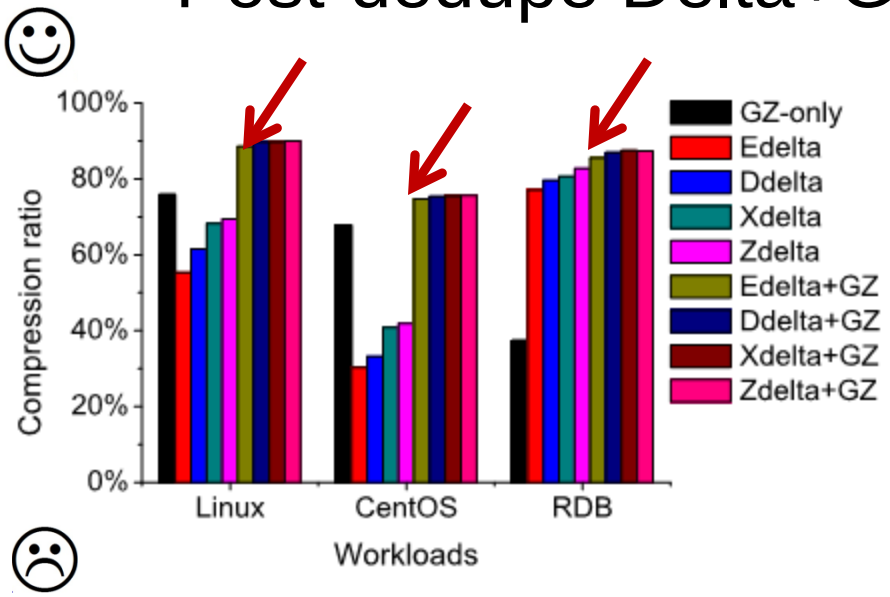


(a) Encoding speed

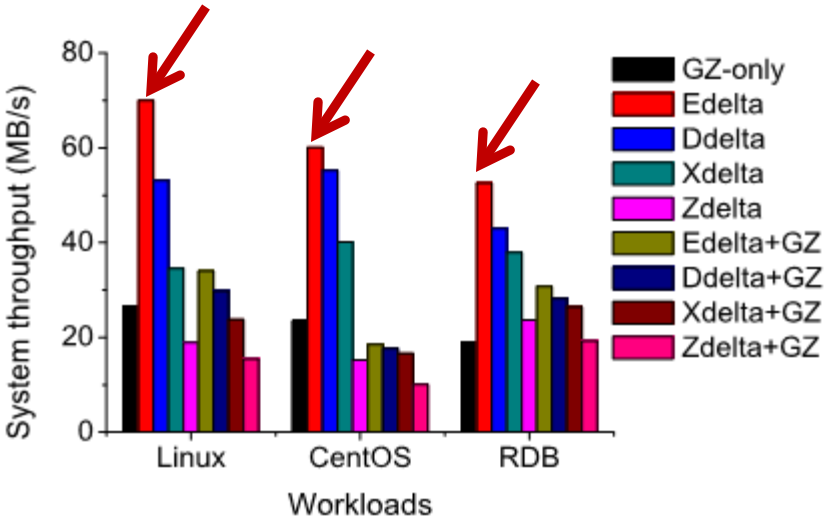
- More than 400MB/s
- 2.5-5X Improv. over Delta
- Not as high as Case Study I
  - Locality missing

The hybrid data reduction system performance

- Post-dedupe Delta+GZ data reduction



(b) Compression ratio



(c) System throughput

Delta+GZIP have the similar compression ratio (Edelta, Xdelta)

Edelta based solutions have the highest system throughputs.

Conclusion and Future Work

- **Edelta is able to delta encode a 4KB-chunk within 2-10 µs**
- **Edelta achieves an encoding speedup of 3-10X over the state-of-the-art DDelta, Xdelta, and Zdelta without noticeably decreasing the compression ratio**

- **Future Work**
  - **Find more promising application scenarios for Edelta**
  - **There are still other bottlenecks for delta compression, such as resemblance detection and reading base chunks/file**

Try to make delta compression “faster” than deduplication

Thanks!

Q & A