dCompaction: Speeding up Compaction of the LSM-Tree via Delayed Compaction, Journal of Computer Science and Technology,

Journal of Computer Science and Technology, 2017

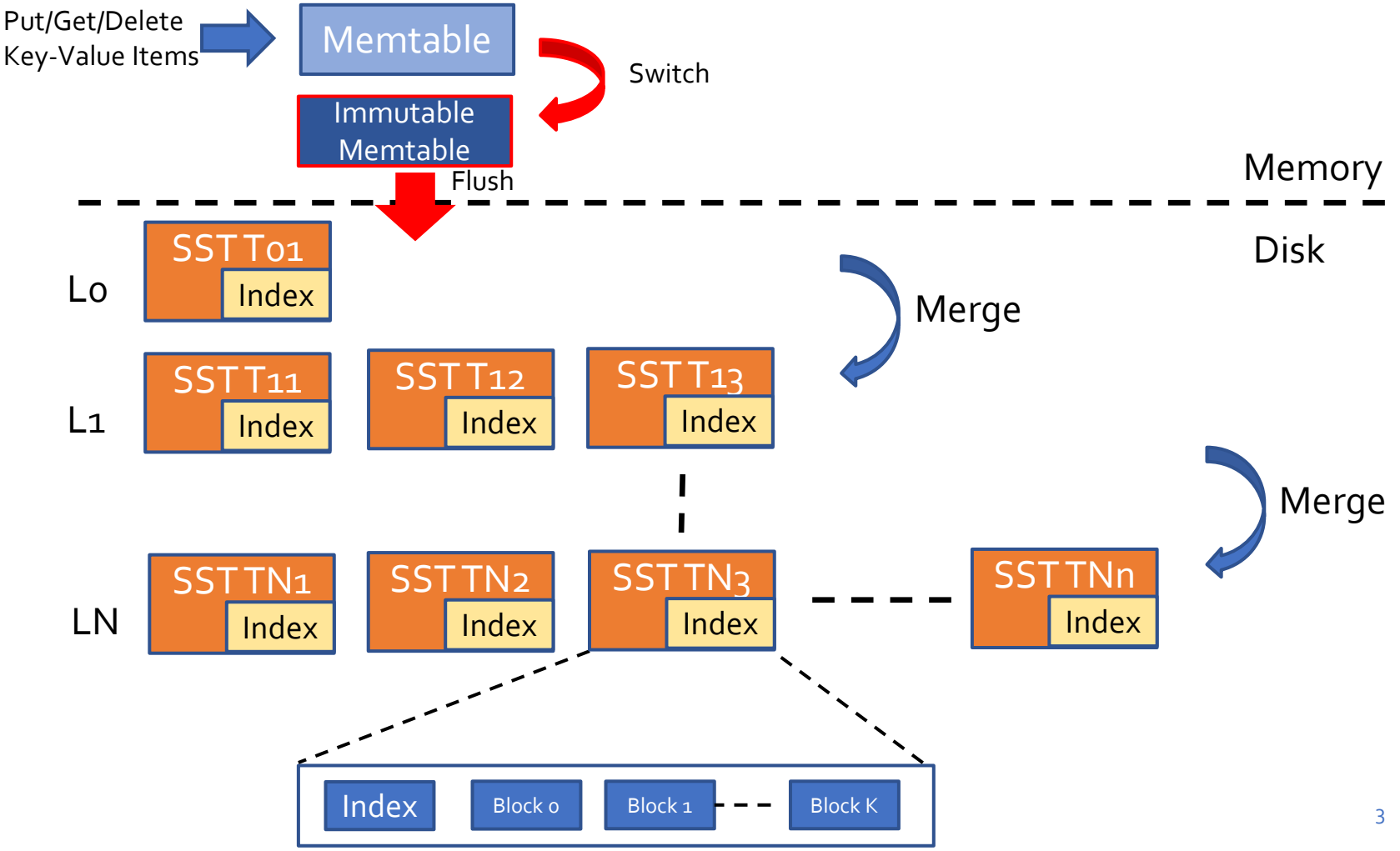
연세대학교 컴퓨터과학과 서주연

과제명: IoT 환경을 위한 고성능 플래시 메모리 스토리지 기반 인메모리 분산 DBMS 연구개발

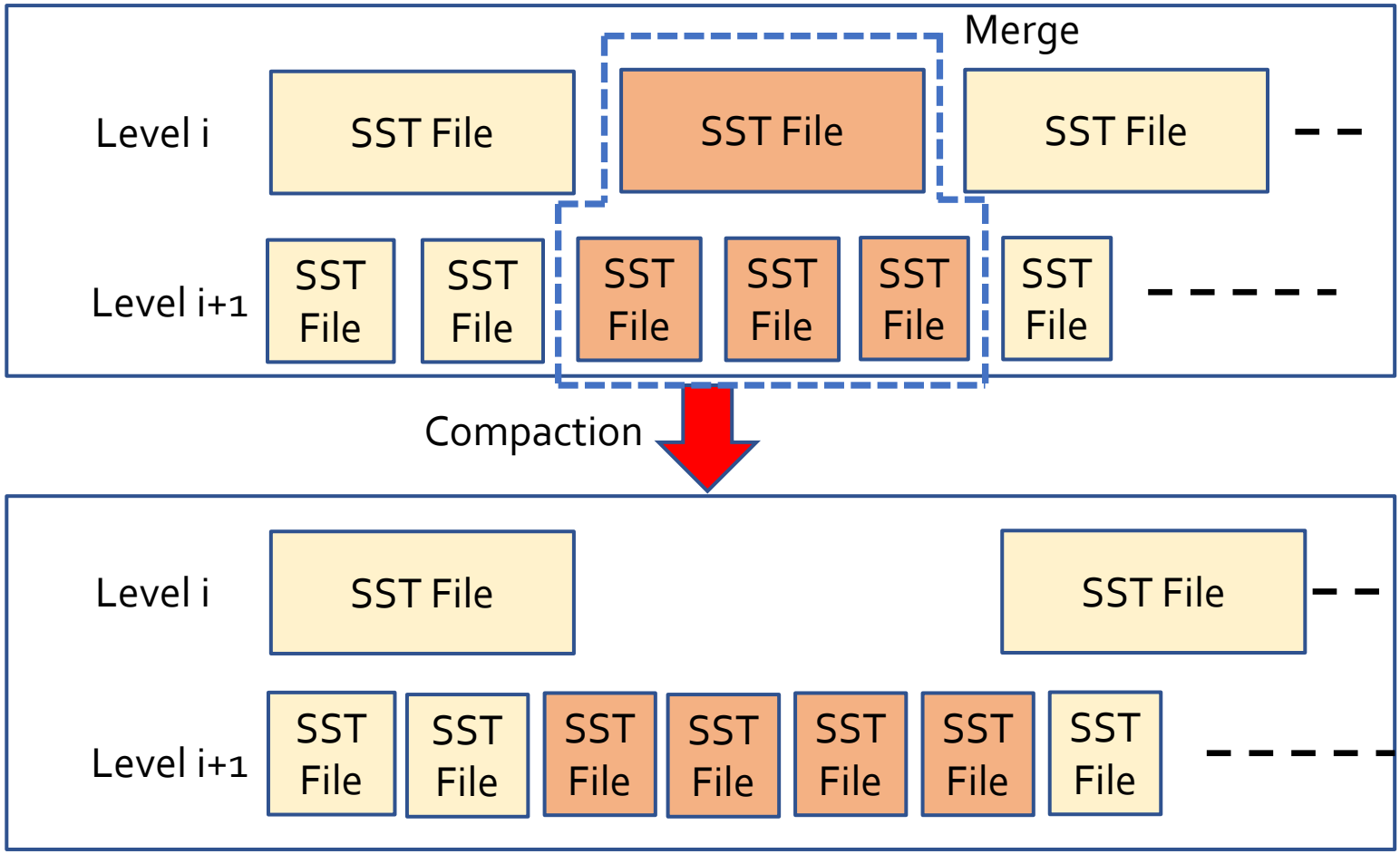
과제번호: 2017-0-00477

Contents

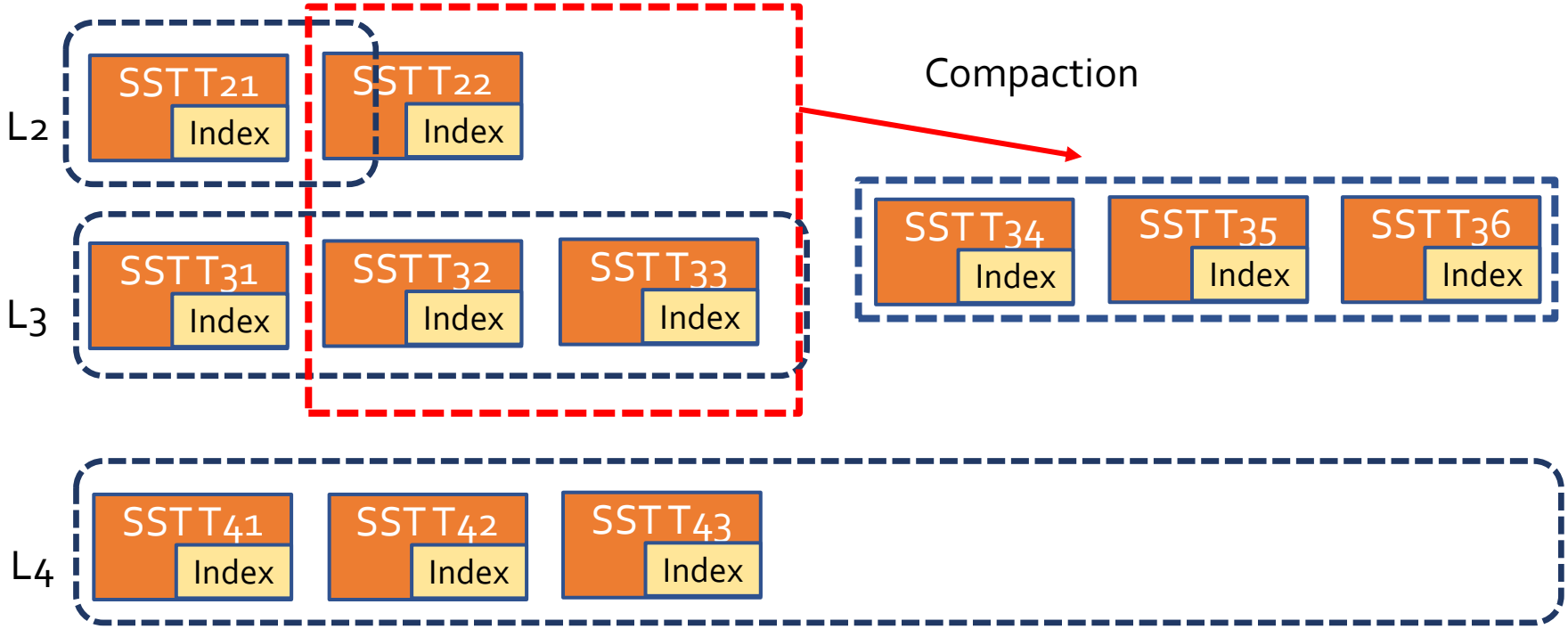
LSM-Tree Data Structure



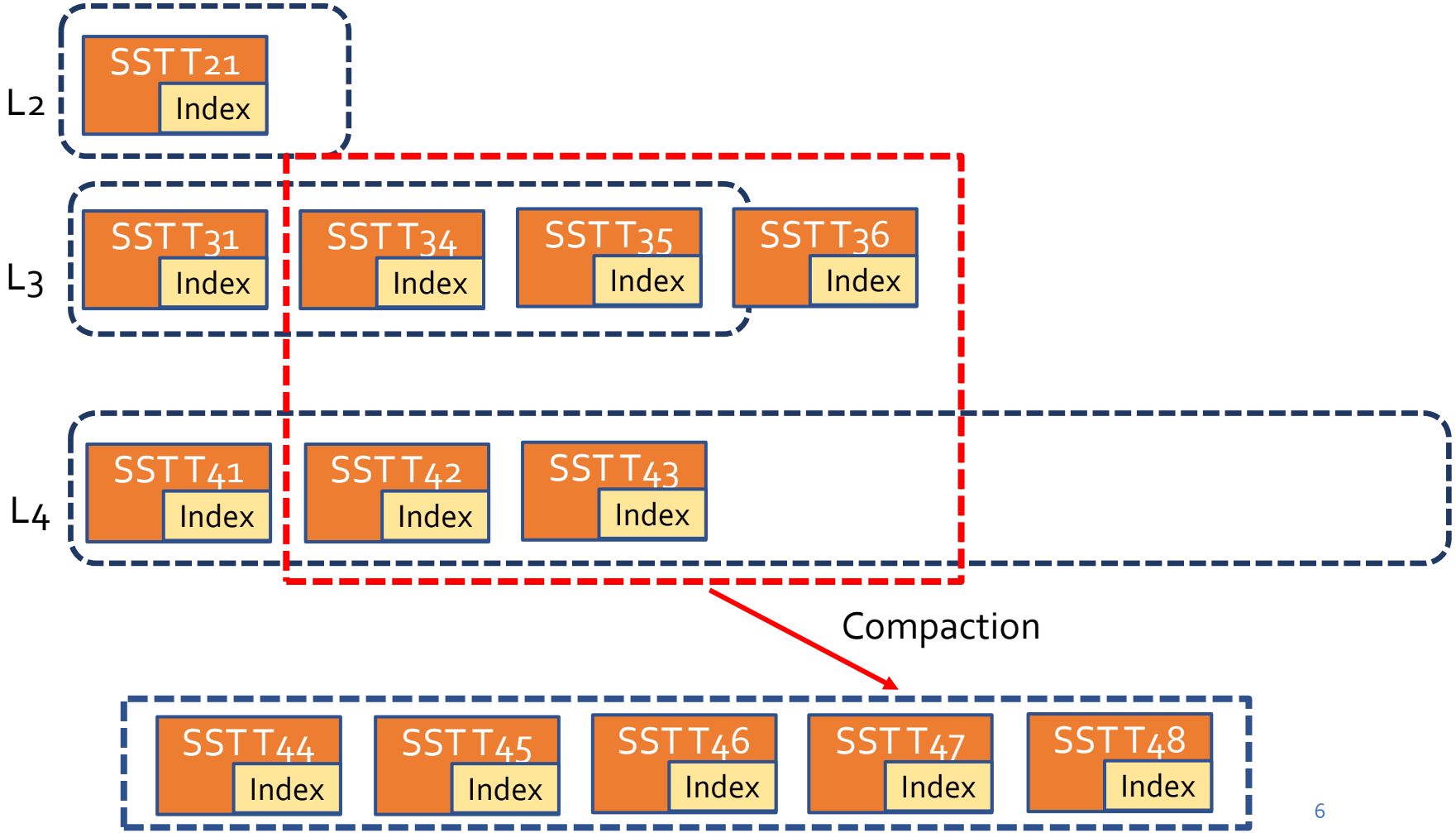
Compaction


Merge
Level i
SST File
SST File
SST File
Level i+1
SST File
SST File
SST File
SST File
SST File
SST File
Compaction
Level i
SST File
SST File
Level i+1
SST File
SST File
SST File
SST File
SST File
SST File
SST File

Compaction Procedure

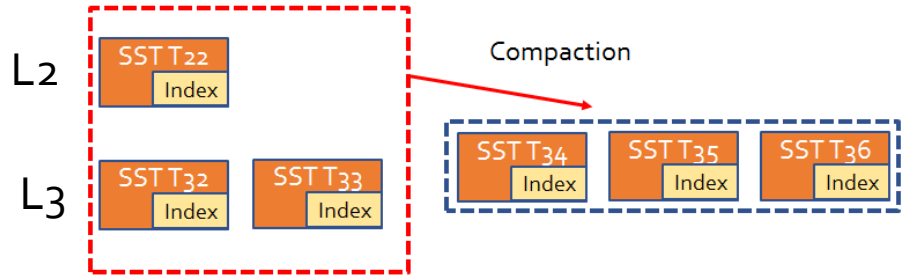


Compaction Procedure

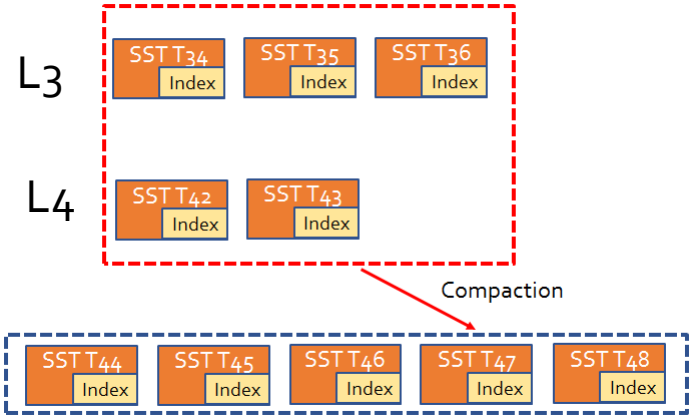


Compaction Problem

- L2->L3 Compaction



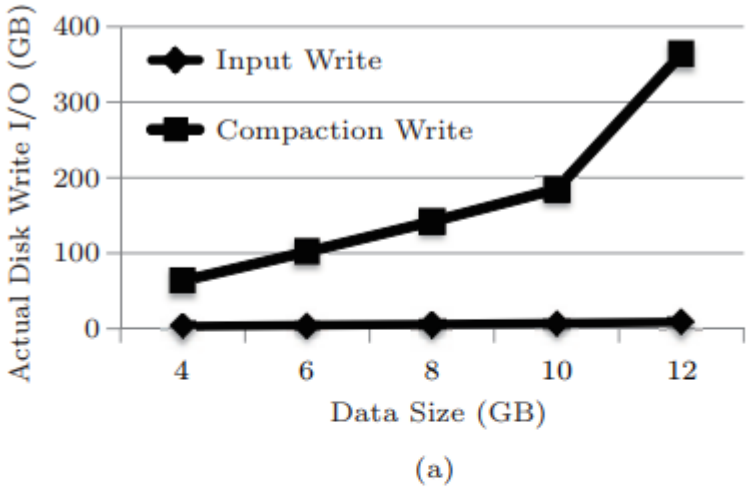
- L3->L4 Compaction



- 3Copy & 3 Write
- 5Copy & 5 Write

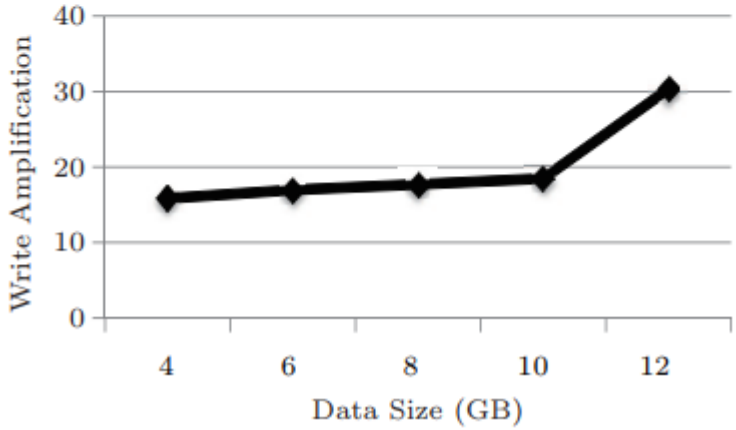
Serious write amplification

Compaction Problem



(a)

Input Write	Compaction Write
4GB	63.39GB
12GB	363.6GB



*Write Amplification Ratio(WAR) = $\frac{Actual\ write\ amount\ to\ disk}{Amount\ of\ data\ written\ by\ users}$

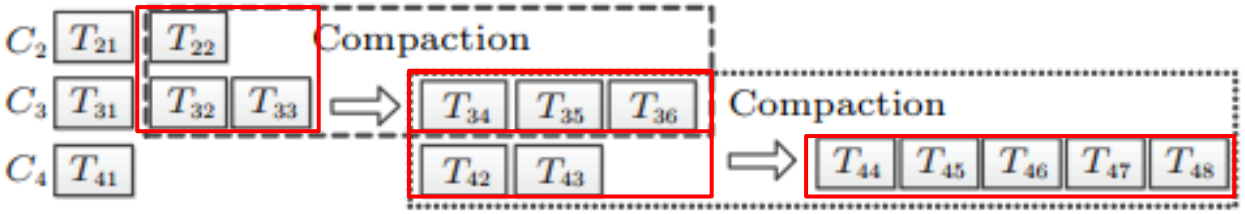
Input Write	WAR
4GB	15.9
12GB	30.3

dCompaction: Speeding up Compaction of the LSM-Tree via Delayed Compaction

---

Feng-Feng Pan, Yin-Liang Yue, Jin Xiong, Journal of Computer Science and Technology, 2017

Motivation

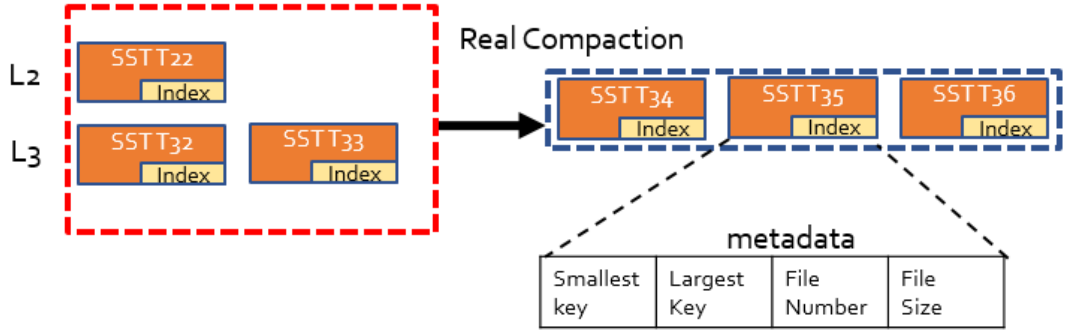


1. KV items in T34, T35, T36 read from T22, T32, T33.
   KV items in T44 ~T48 read from T34, T35, T36, T42, T43

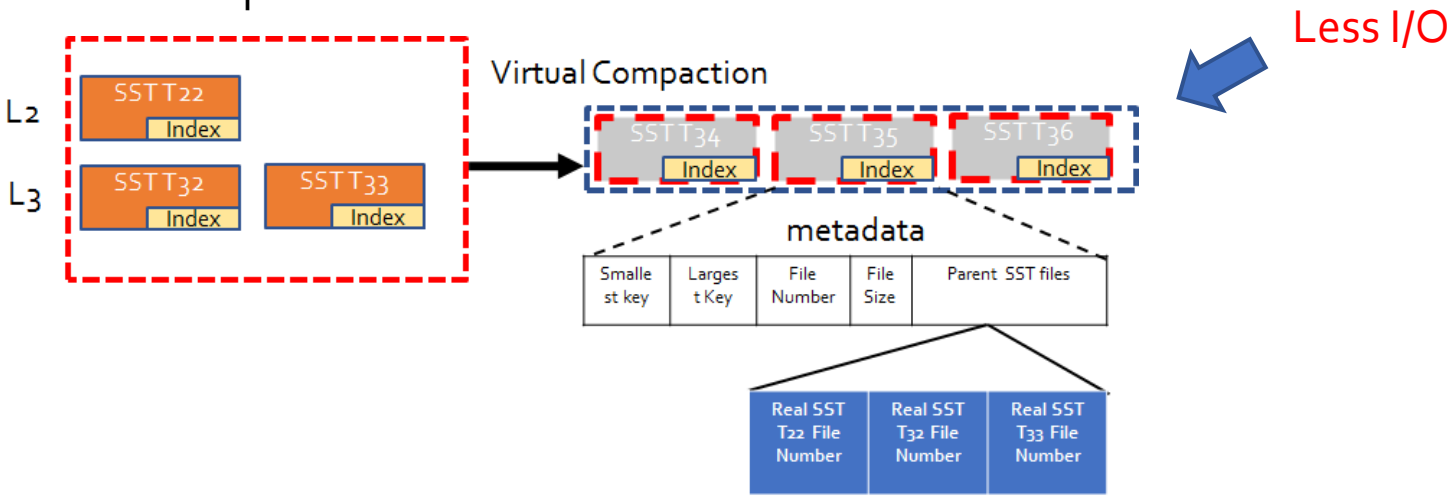
2. Repeated reads and writes during compaction procedures.

Main Idea

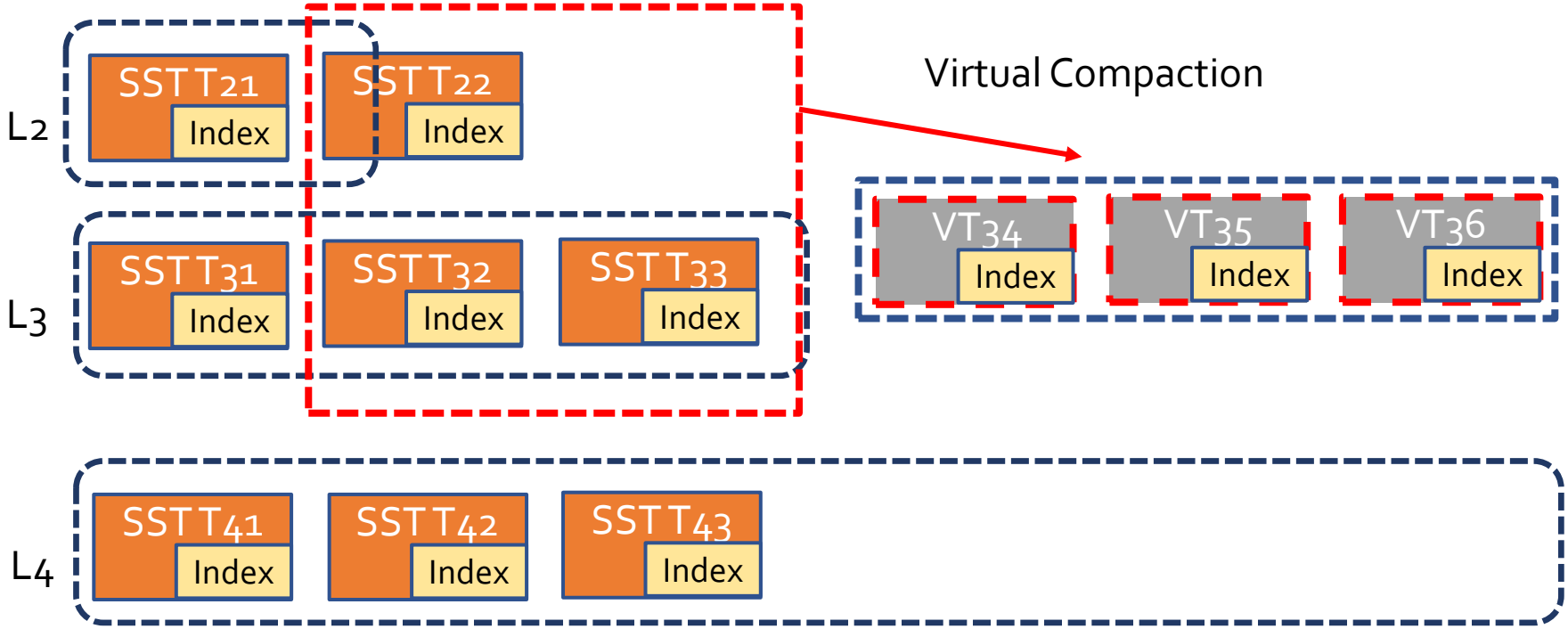
- Real Compaction



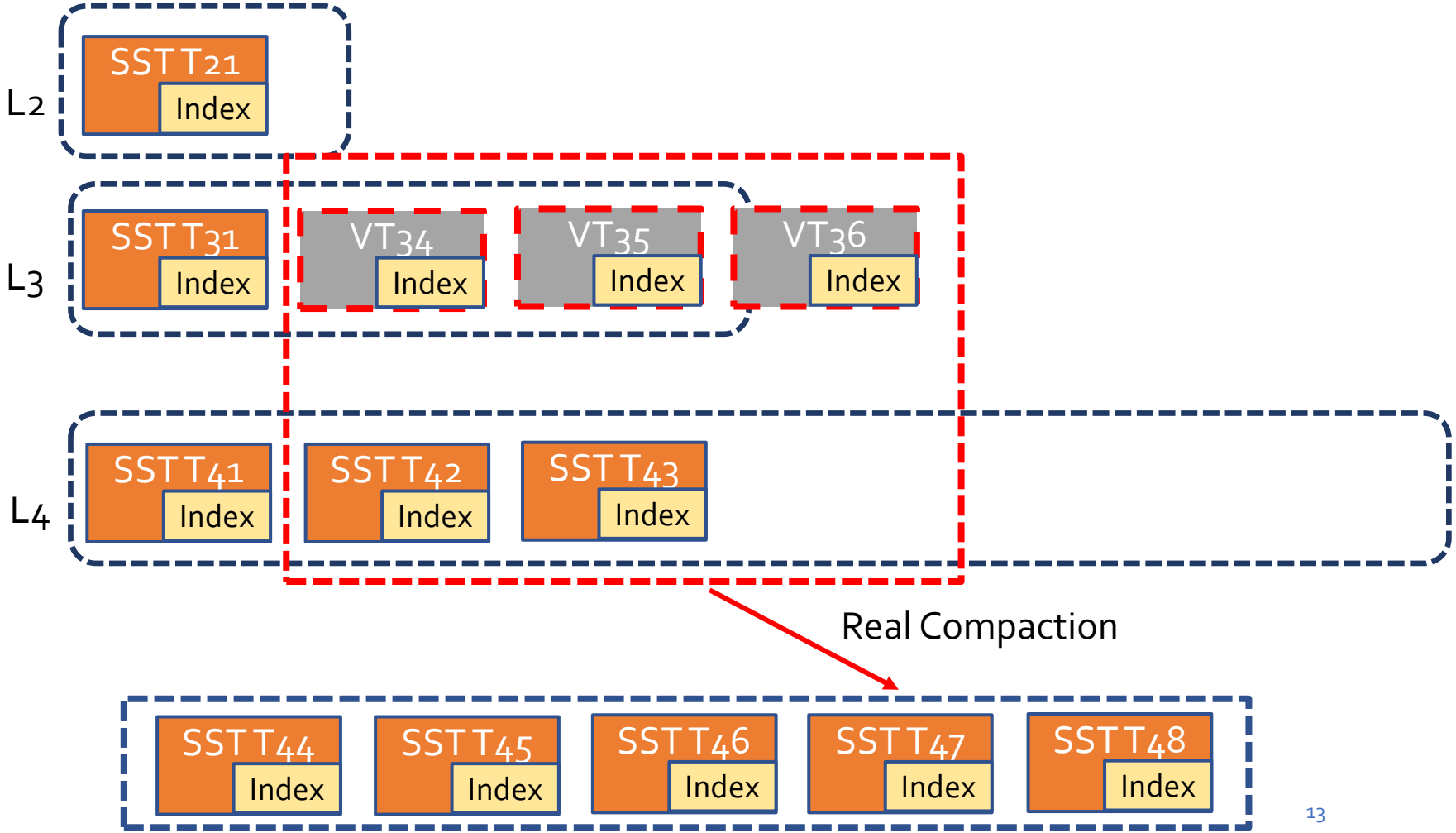
- Virtual Compaction



dCompaction Procedure

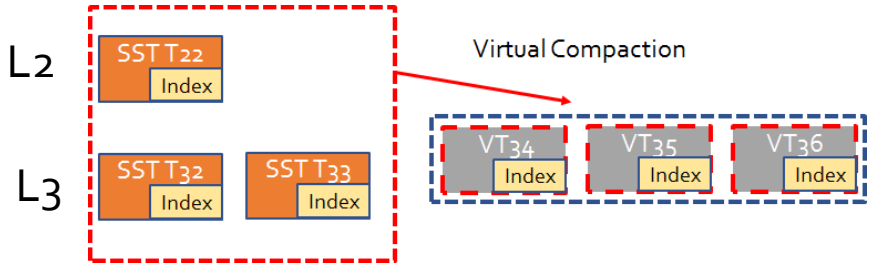


dCompaction Procedure

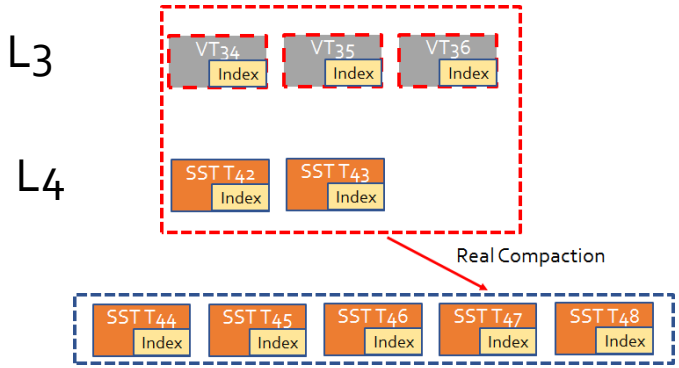


dCompaction

- L2->L3 Compaction



- L3->L4 Compaction

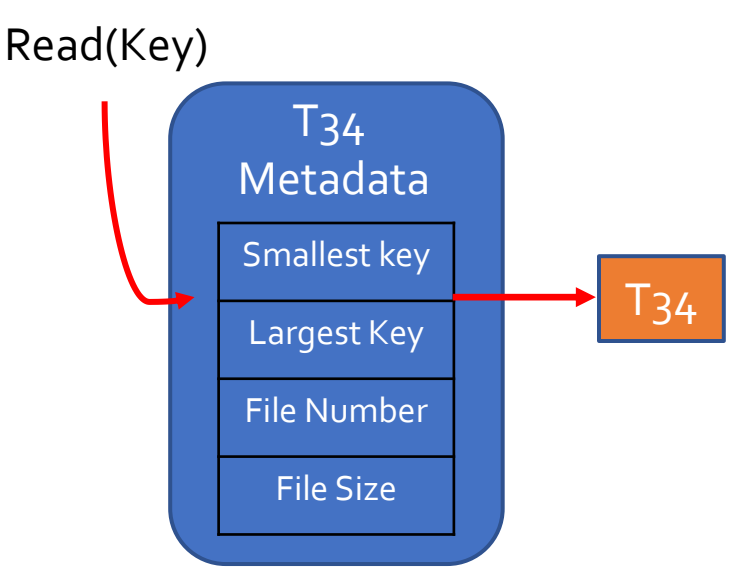


- 3Copy & 0Write
- 5Copy & 5 Write

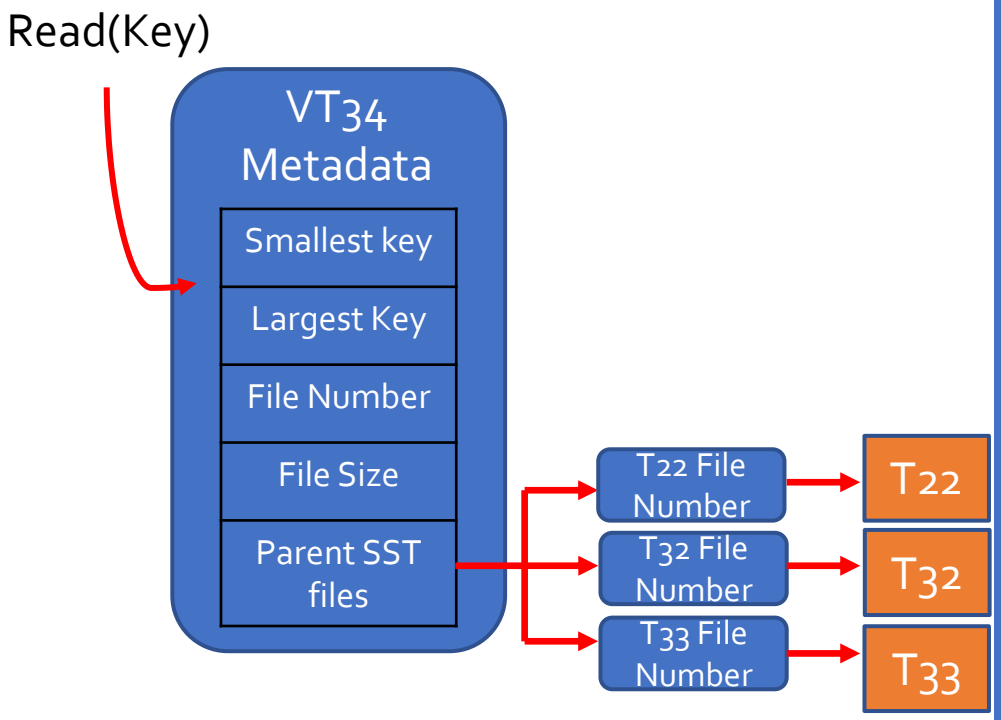
Less Disk I/O

Read Process

- Real SST Files



- Virtual SST Files



Trigger condition: VCT

- Virtual Compaction Threshold : threshold to trigger either virtual compaction or real compaction

- Algorithm

```
Compaction starts;
Choose SSTables needing to be compacted;
Count the number of real SSTables in this compaction;
Count = the number of real SSTables;
if Count < VCT then
  | Trigger the virtual compaction;
else
  | Trigger the real compaction;
end
```

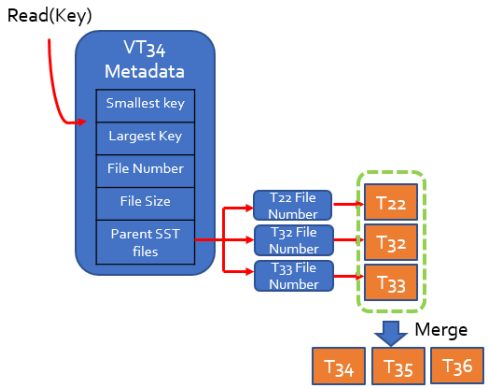
  - VCT = 0 : Real compaction
  - VCT ↑ : Frequency of virtual compaction ↑

Trigger condition: VSMT

- Virtual SST files merge threshold : the number of real SST Files which is the virtual SST File is derived from.

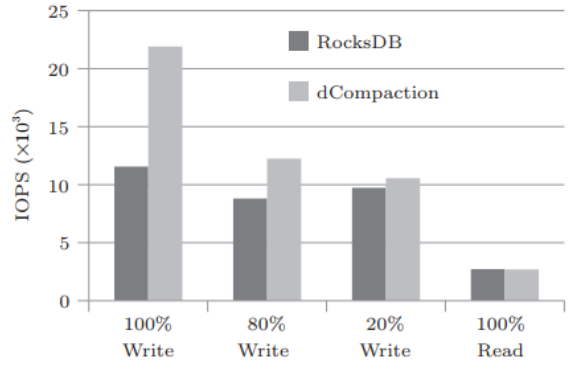
- Algorithm

```
if SSTable.type == virtualSSTable then
    Count the number of real SSTables in the virtual SSTable;
    Count = the number of real SSTables;
    if Count ⩾ VSMT then
        Merge virtual SSTable into real SSTables;
    end
end
```

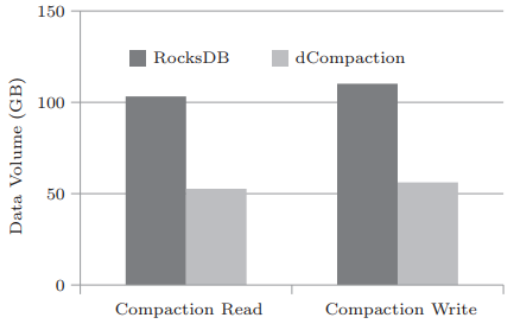


- Reduce lookup I/O & Improve Read performance
- Small VSMT : Read performance ↑ but Merge frequency ↑
- Large VSMT : Read performance ↓ but Merge frequency ↓

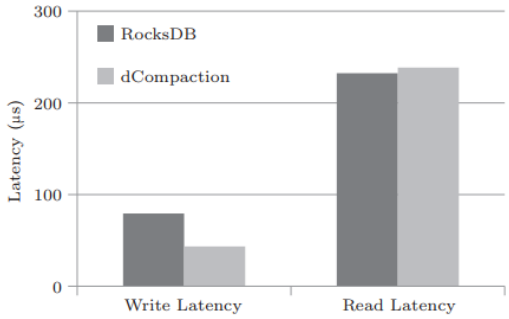
Evaluation- YCSB



- 100% Write - dCompaction improves by 89.47%
  → virtual compaction reduce I/O overhead



- dCompaction I/O saving
  - Read - 48.27%
  - Write- 51.23%



- Lower Write Latency But Higher Read Latency

Thanks