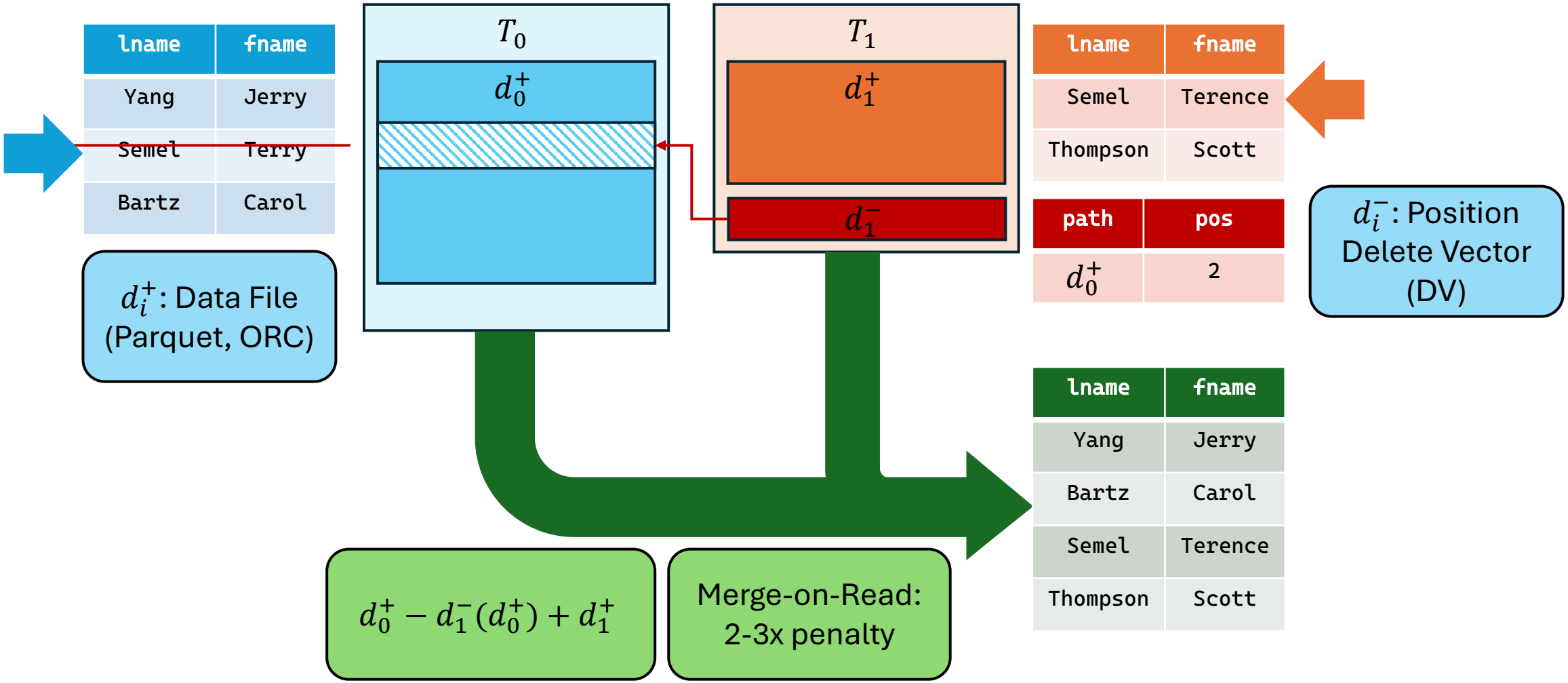
Commutative Compaction

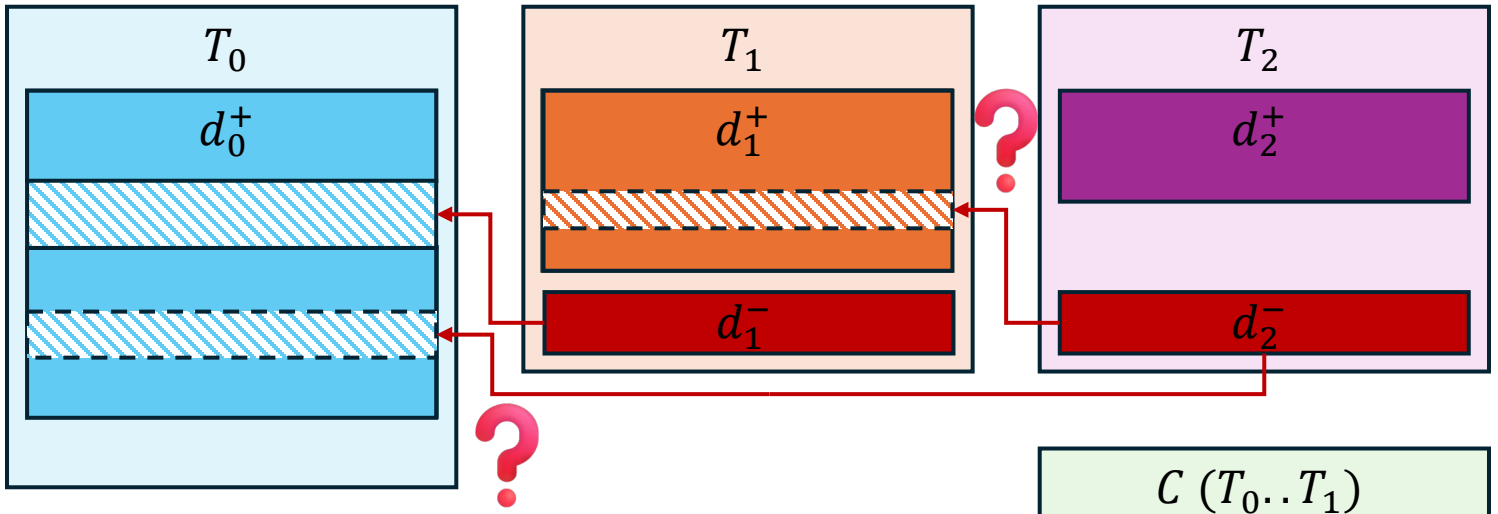
Chris Douglas, Joseph M. Hellerstein

UC Berkeley

Compaction

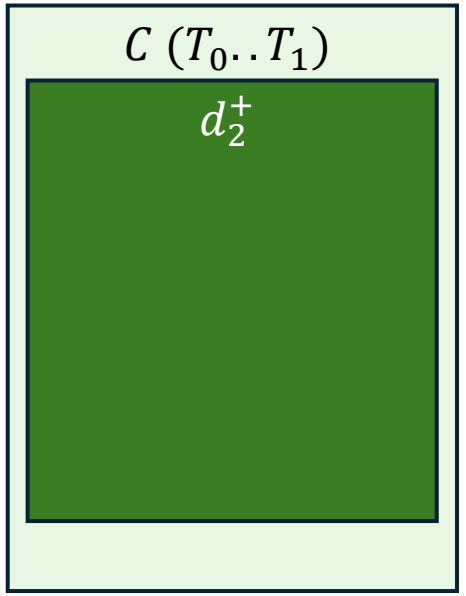

lname
fname
Yang
Jerry
Semel
Terry
Bartz
Carol
d_i^+: Data File (Parquet, ORC)
T_0
d_0^+
T_1
d_1^+
d_1^-
lname
fname
Semel
Terence
Thompson
Scott
path
pos
d_0^+
2
d_i^-: Position Delete Vector (DV)
$d_0^+ - d_1^-(d_0^+) + d_1^+$
Merge-on-Read: 2-3x penalty
lname
fname
Yang
Jerry
Bartz
Carol
Semel
Terence
Thompson
Scott

Compaction

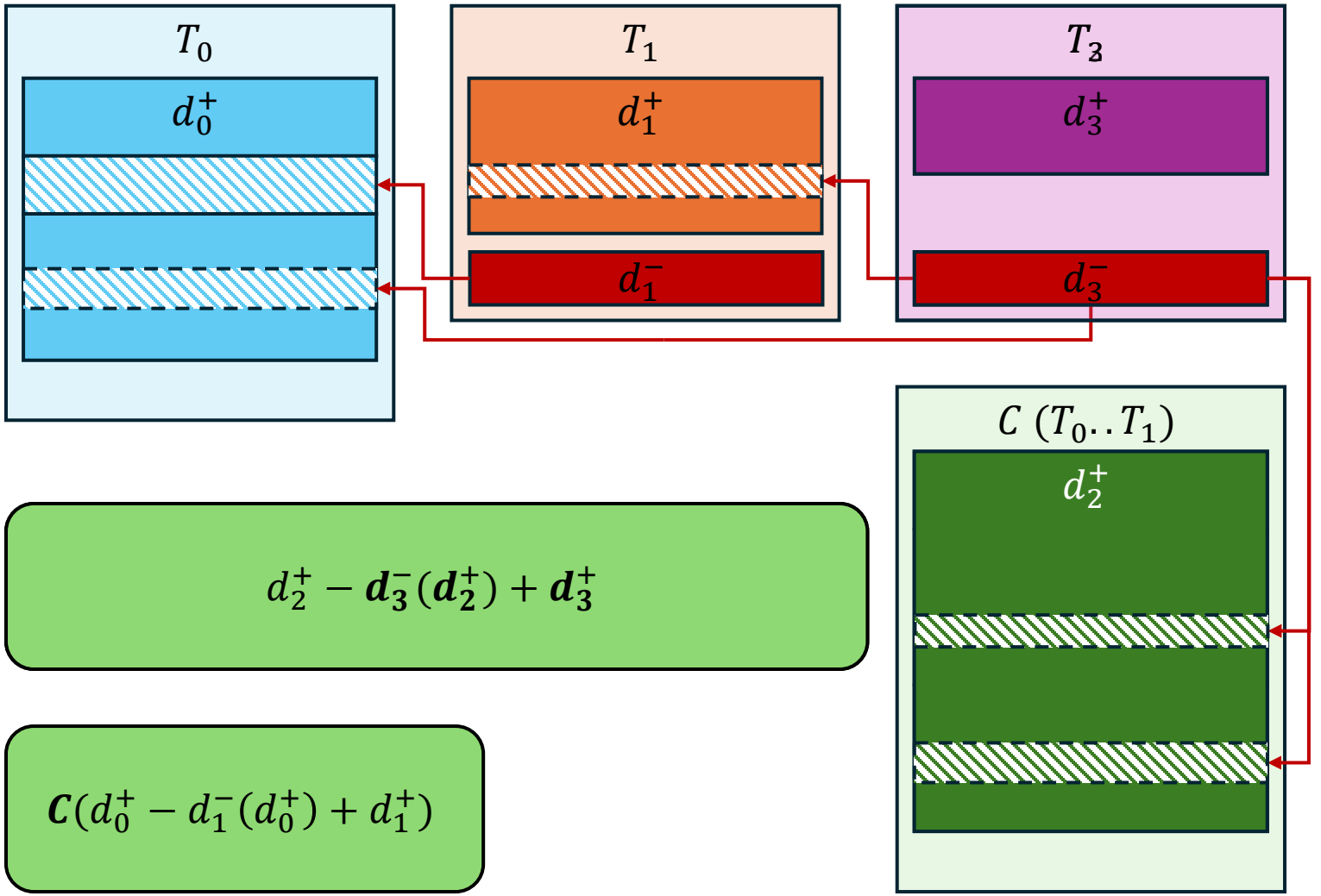


$$d_0^+ - d_1^-(d_0^+) - \boldsymbol{d_2^-(d_0^+)} + d_1^+ - \boldsymbol{d_2^-(d_1^+)} + \boldsymbol{d_2^+}$$

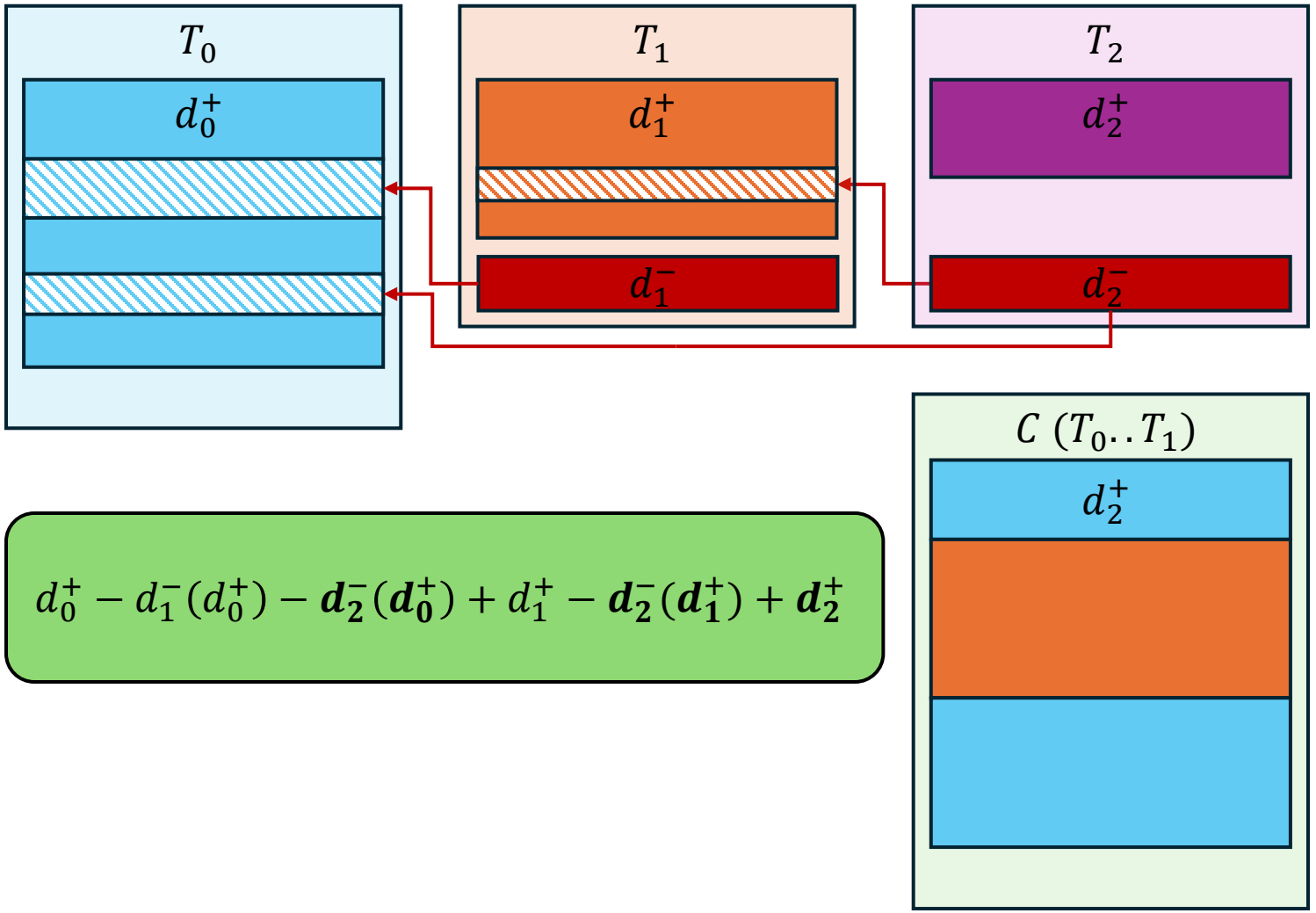
$$\boldsymbol{C}(d_0^+ - d_1^-(d_0^+) + d_1^+)$$



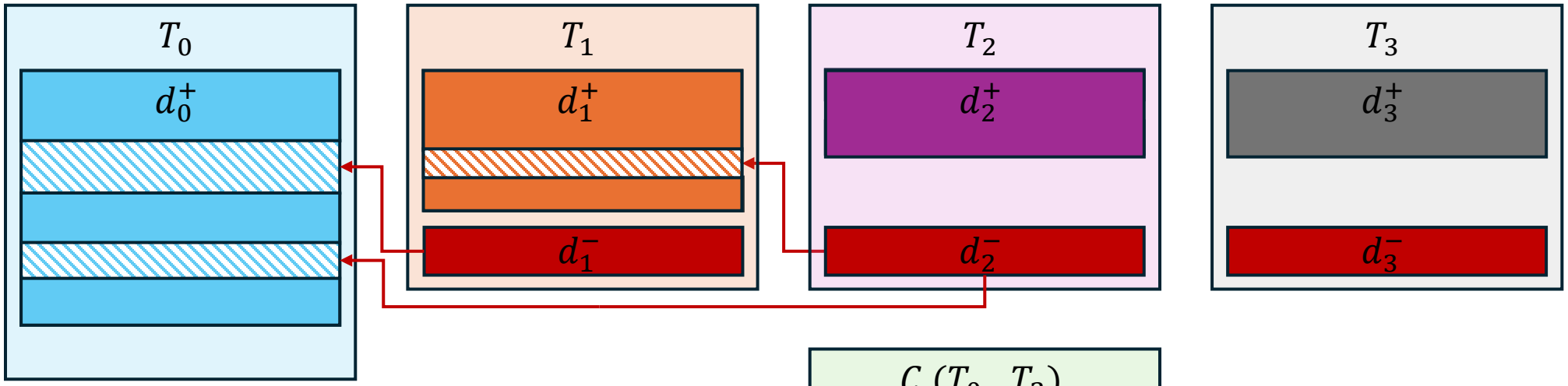
Compaction

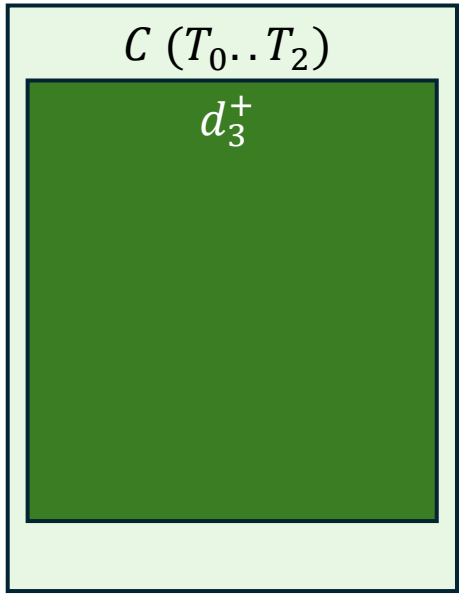

T_0
d_0^+
T_1
d_1^+
d_1^-
T_3
d_3^+
d_3^-
$C\ (T_0..T_1)$
d_2^+
$d_2^+ - \boldsymbol{d_3^-(d_2^+)} + \boldsymbol{d_3^+}$
$\boldsymbol{C}(d_0^+ - d_1^-(d_0^+) + d_1^+)$

Compaction


T_0
d_0^+
T_1
d_1^+
d_1^-
T_2
d_2^+
d_2^-
$C\ (T_0..T_1)$
d_2^+
$d_0^+ - d_1^-(d_0^+) - \boldsymbol{d_2^-(d_0^+)} + d_1^+ - \boldsymbol{d_2^-(d_1^+)} + \boldsymbol{d_2^+}$

Compaction





deref ➡️

does *not* commute

reorder_rows 🔀

$$d_0^+ - d_1^-(d_0^+) - \boldsymbol{d_2^-(d_0^+)} + d_1^+ - \boldsymbol{d_2^-(d_1^+)} + \boldsymbol{d_2^+}$$

$$\boldsymbol{C}(d_0^+ - d_1^-(d_0^+) - d_2^-(d_0^+) + d_1^+ - d_2^-(d_1^+) + d_2^+)$$

Layout Conflicts

Logical noops are conflicts because compactions invalidate *direct references*

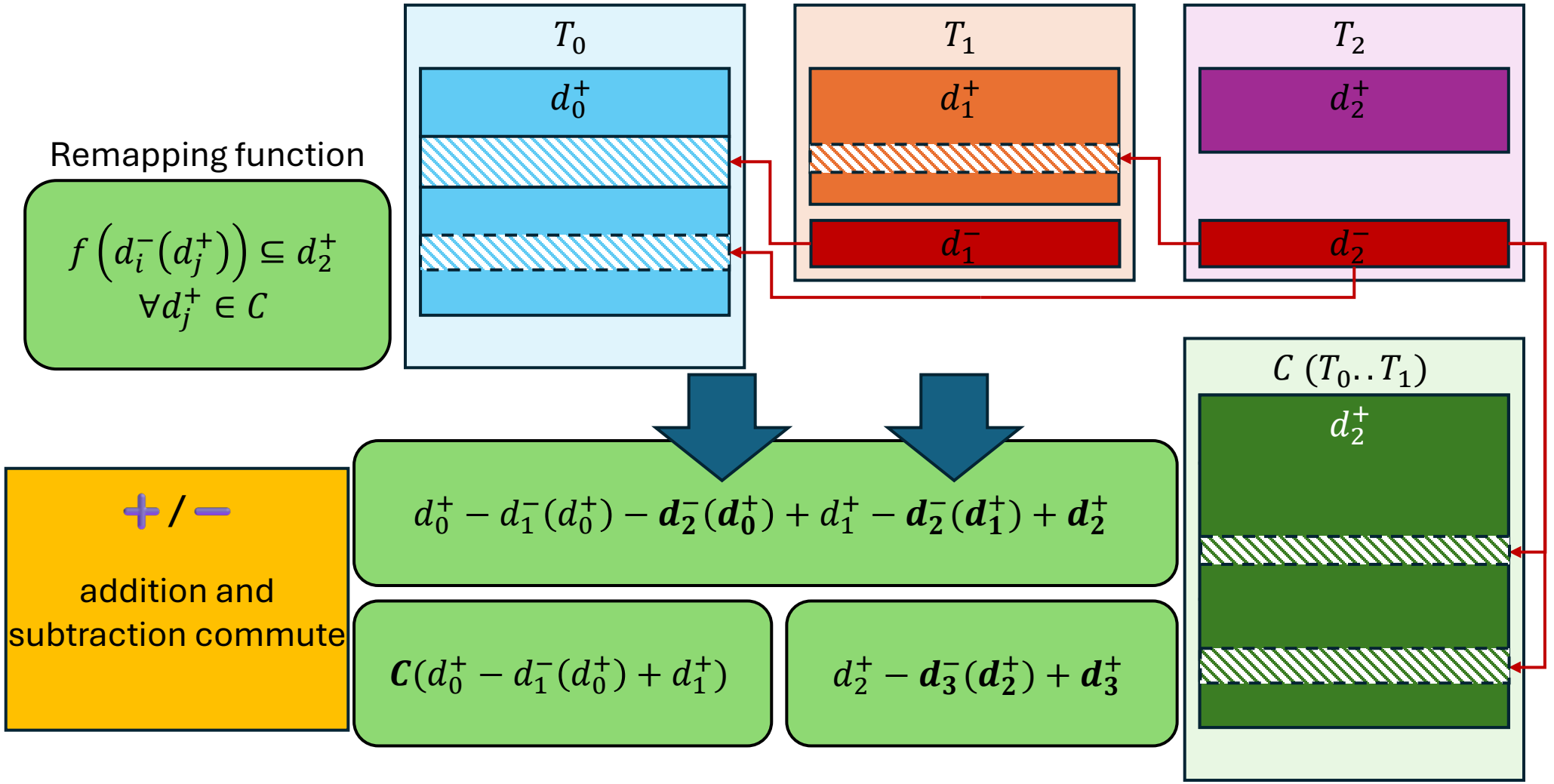
Pointer Swizzling? (rewrite ➡️)

- ❌ Objects are immutable
- ❌ No registry of active readers
- ❌ Write amplification

Intercept references? (🔀 ➡️ ➡️)

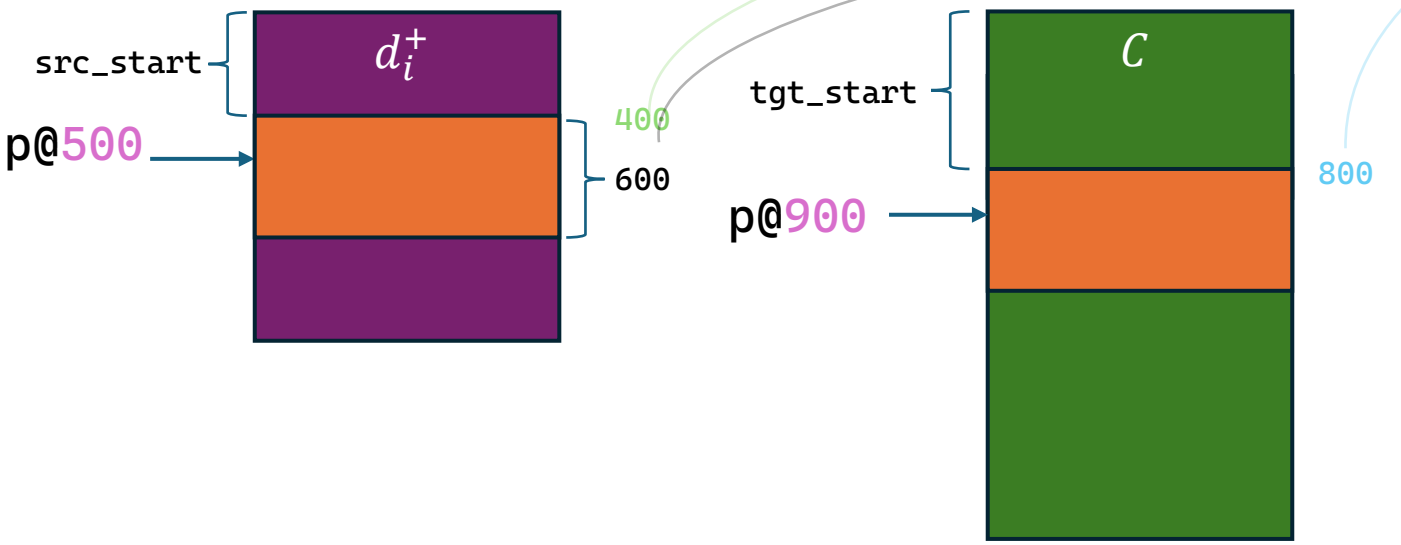
- ❌ Provision a separate service
- ❌ Full: HA+bandwidth requirements
- ❌ Ref: Data races/stateful

Compaction: Layout Conflicts



Compaction Maps

- Definition: set of *remapping functions*
- Intuition: contiguous ranges remapped/deleted
- Compact: 2.6KiB for 10 runs and 8.9KiB for 10k runs



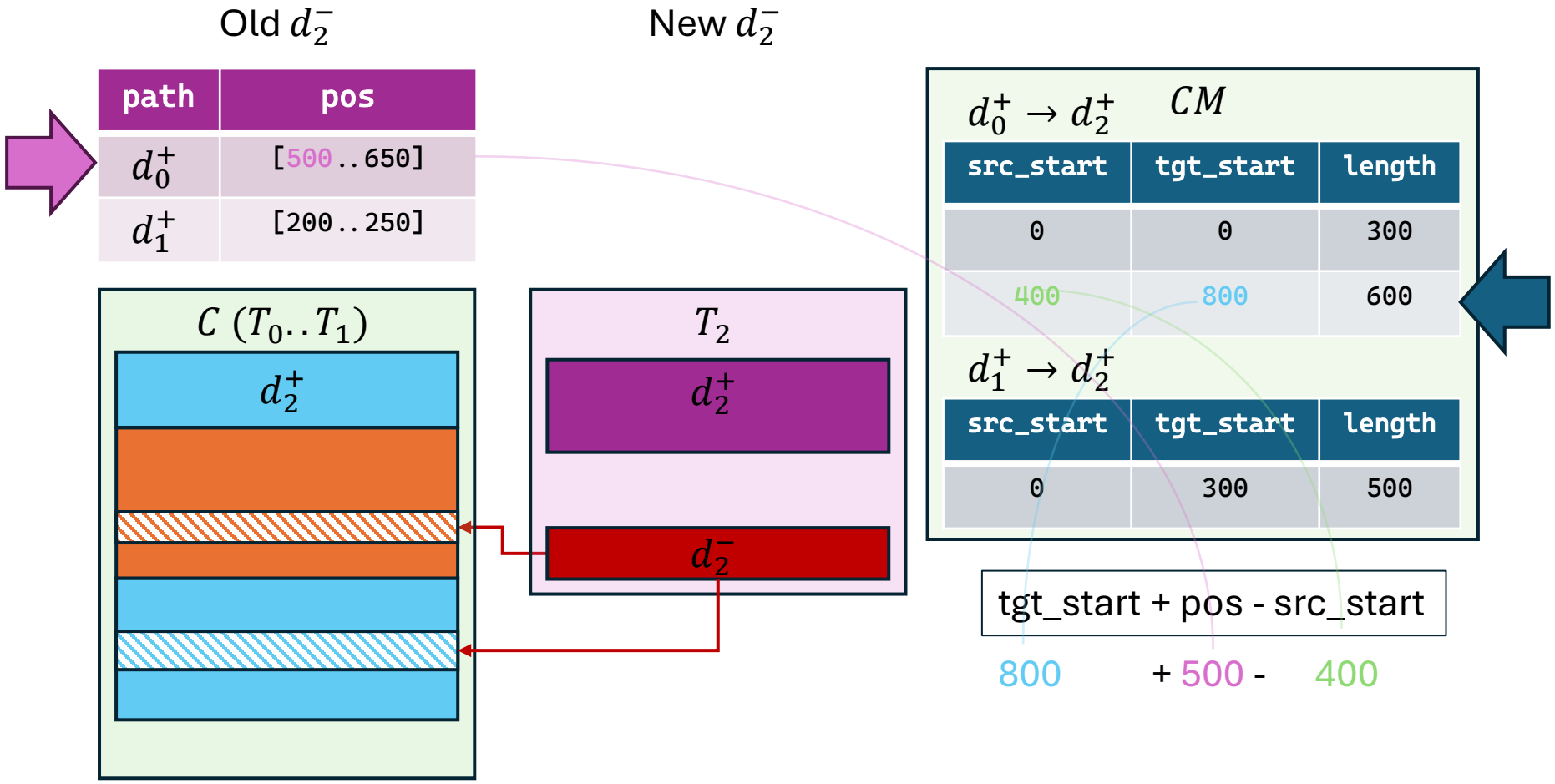
$d_0^+ \rightarrow d_2^+$ CM

src_start	tgt_start	length
0	0	300
400	800	600

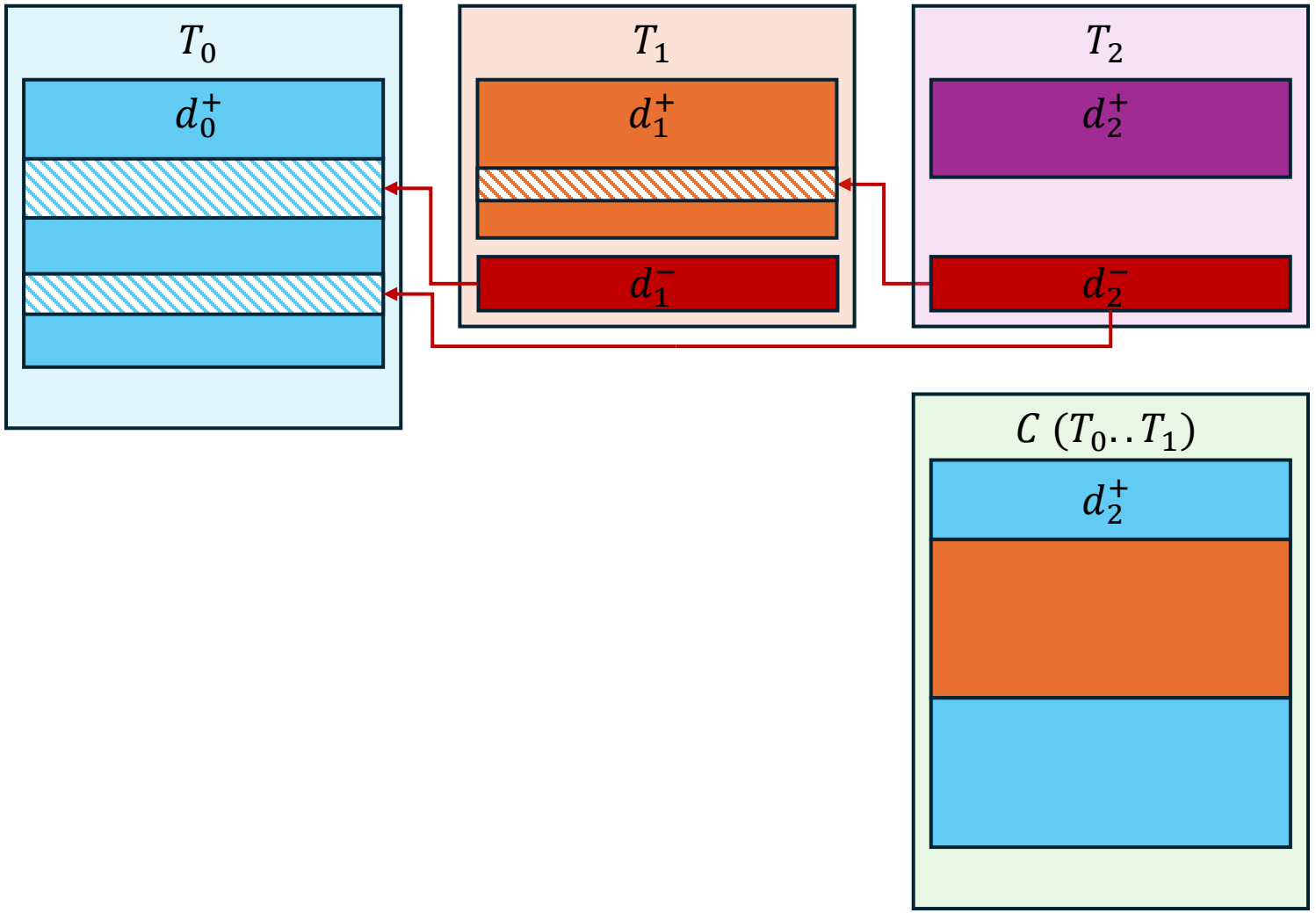
$d_1^+ \rightarrow d_2^+$

src_start	tgt_start	length
0	300	500

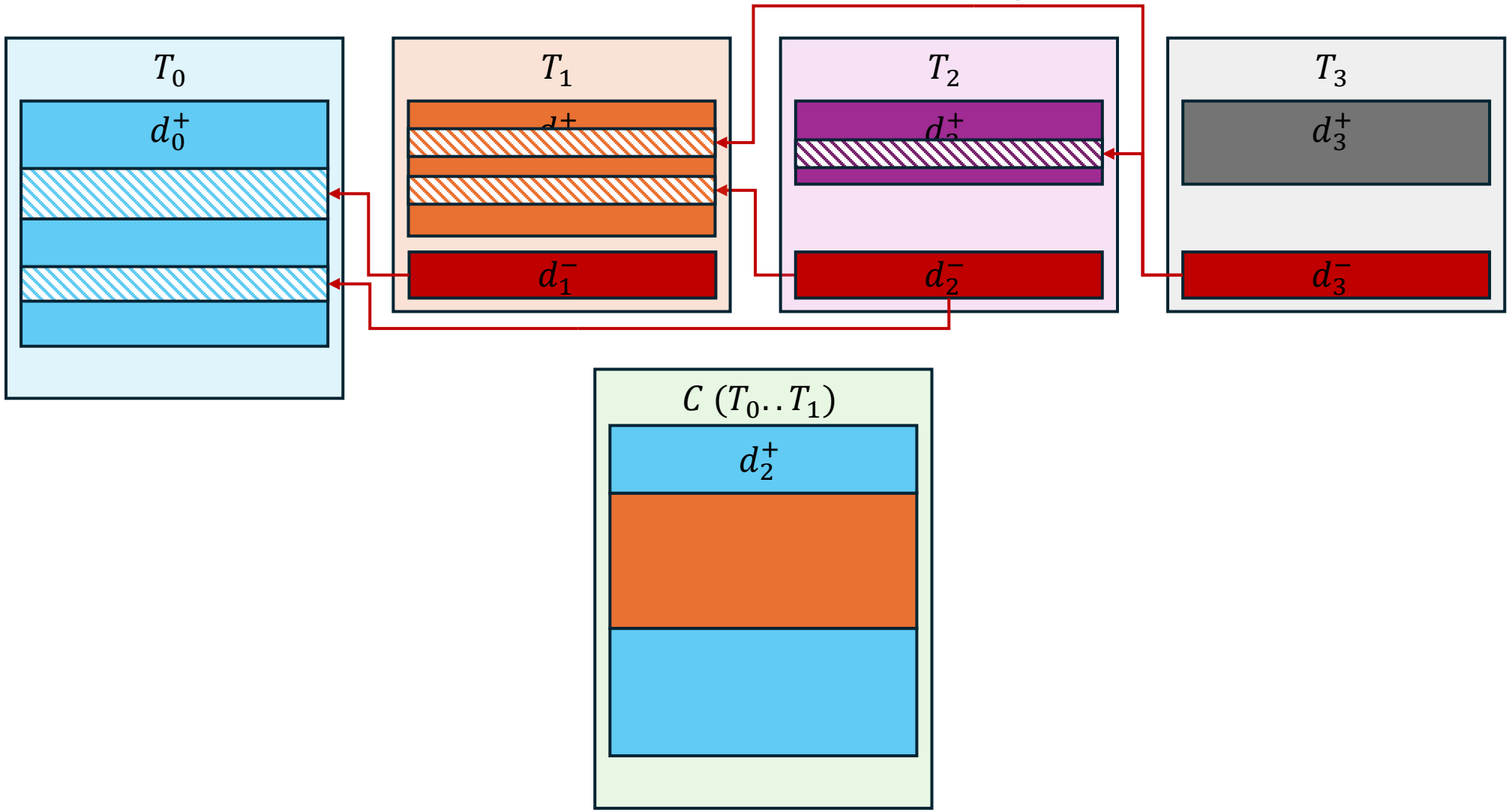
Commute T_2 with C



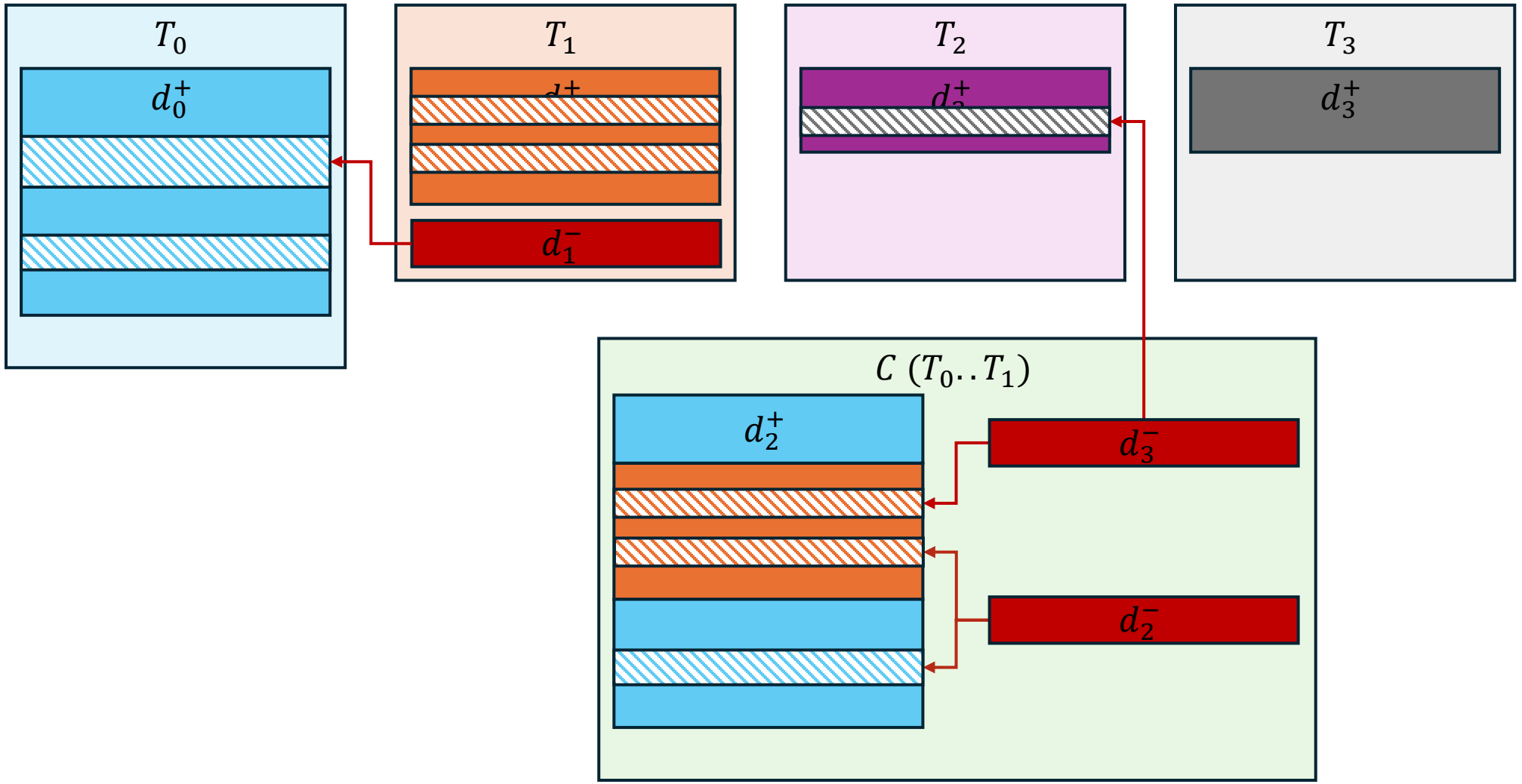
Commute C with T_2



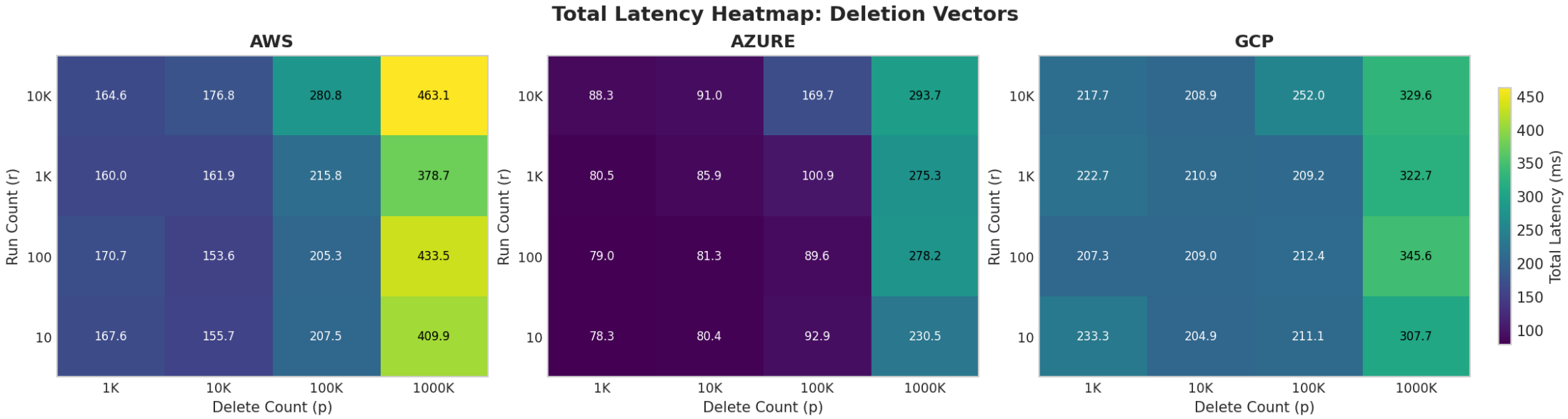
Commute C with T_2... and T_3



Commute C with T_2... and T_3

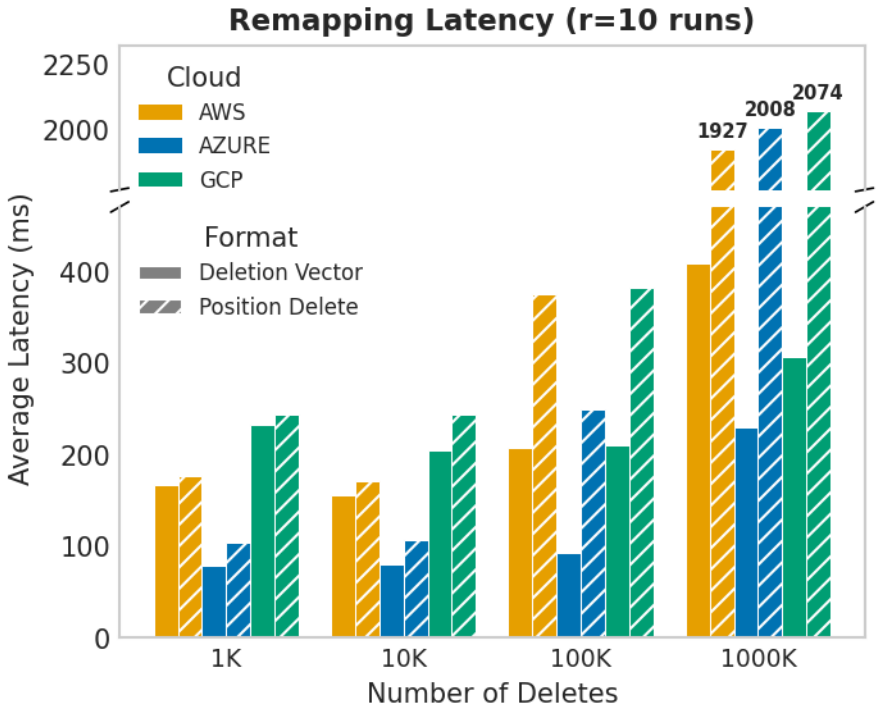


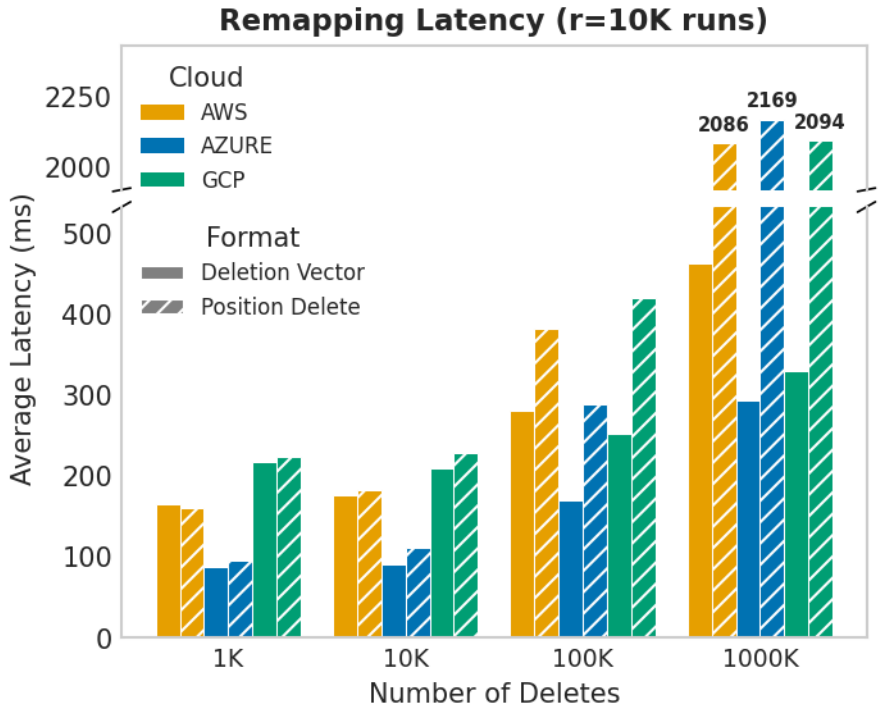
Remapping Latency: Iceberg v3 (DV)



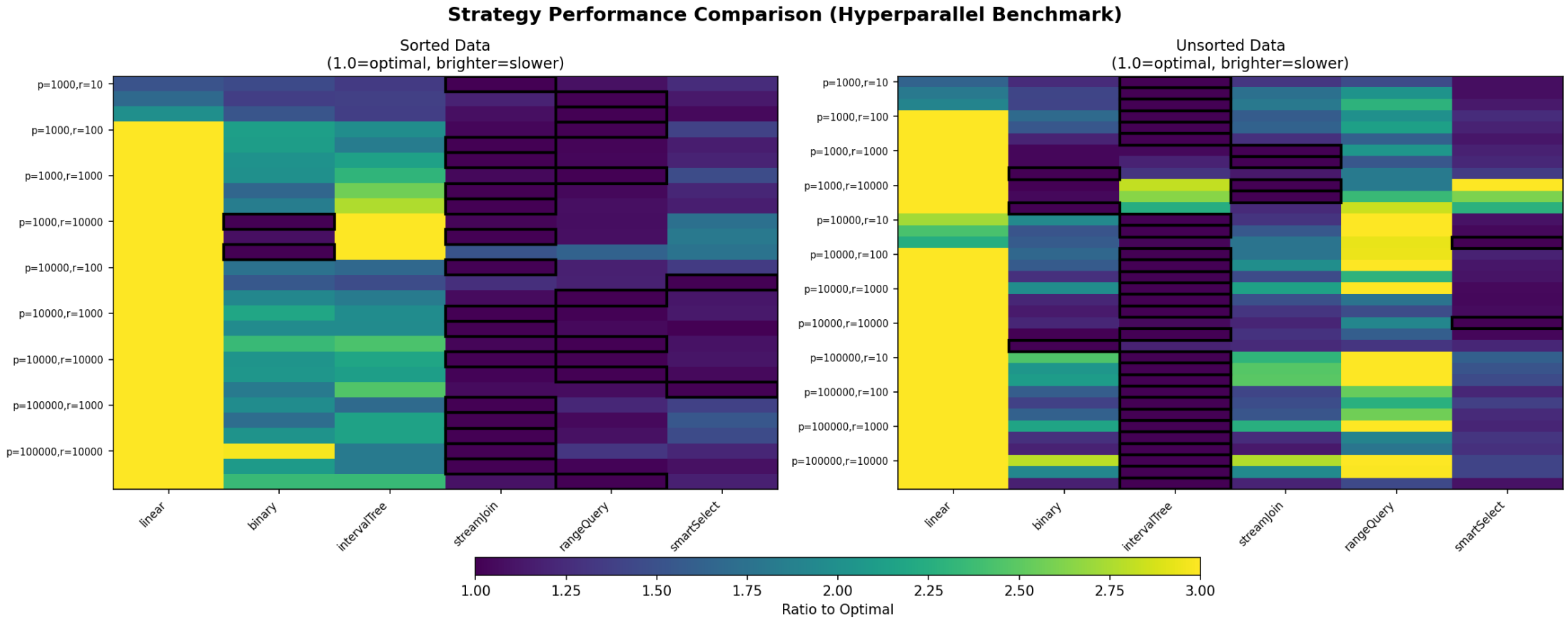
Re-running compaction: 1-5 minutes
7200x-360000x speedup

Remapping Latency: end-to-end





Remapping: Policy



Future Work

- SERIALIZABLE detects spurious read conflicts
- SORT, Z-ORDER compactions
  - Too large? Change granularity of the map?
- Other maintenance: deduplication, GDPR
  - Rebase applications that do their own dedup?
- Merge index (REMIX, Zhong et al. FAST'21)
  - "Anchor keys" improve lookup + scan without a compaction
- Checkpoint inversion
  - Rewrite transaction snapshots to reference newer checkpoints

Table Formats and Data Warehouses

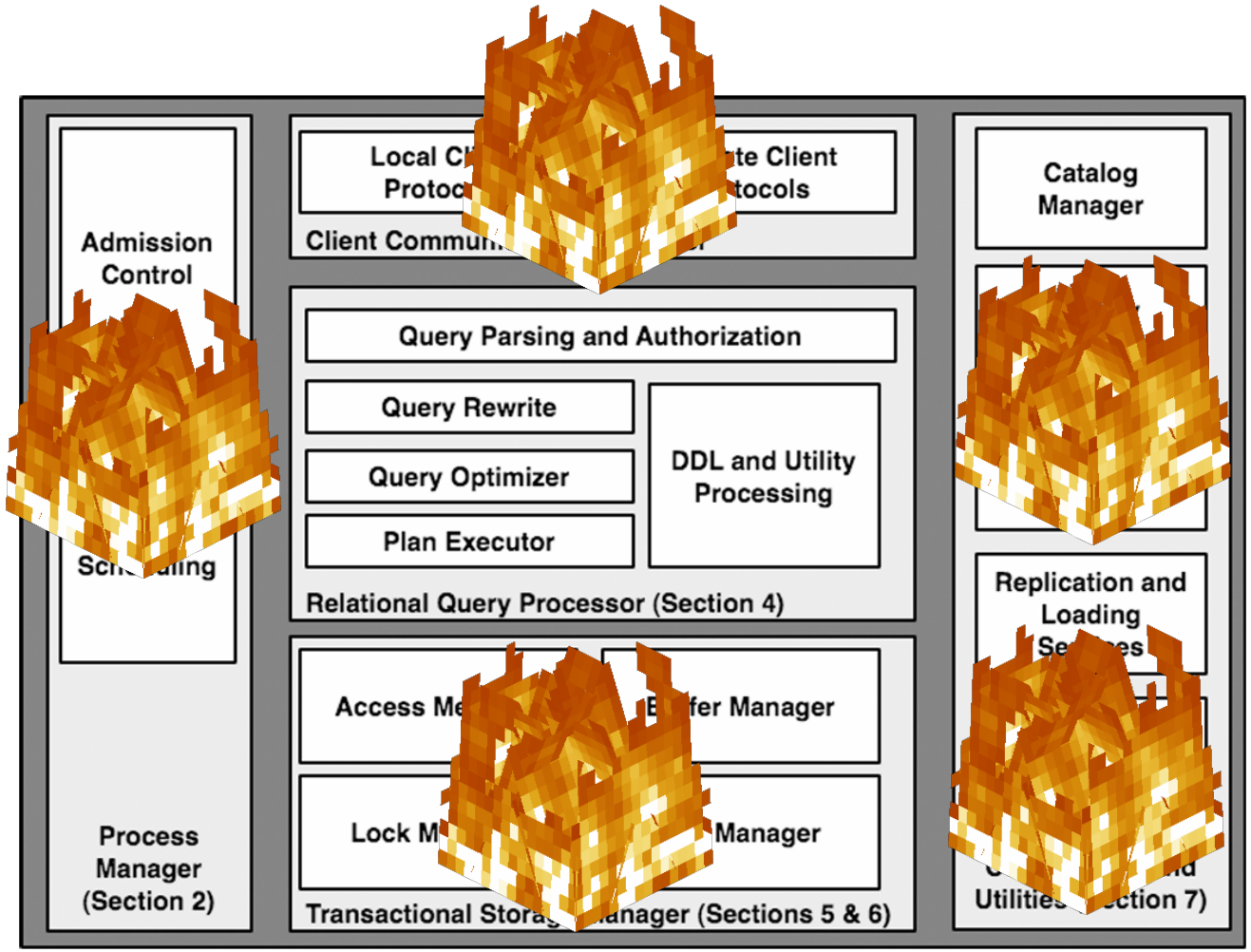


Table Formats and Data Warehouses

- Open world
  - No knowledge of concurrent transactions
  - No admission control
  - No external coordination
- *Transaction eversion*
  - Execution in storage
  - Write recovery data into the table assuming concurrent updates

Credit: Linda Thomas-Fowler
https://www.flickr.com/photos/ac4lt/

Thank you

- Avoid re-execution of compactions and transactions
- Writer-centric state to *repair* conflicts

Chris Douglas
chris_douglas@berkeley.edu

Joseph M. Hellerstein
hellerstein@berkeley.edu

Thank you
Ashvin Agrawal, Carlo Curino, Owen O'Malley, Tiemo Bang, Gopal Vijayaraghavan, Natacha Crooks, Micah Murray, Pavan Lanka, and Sumedh Sakdeo

Backup slides

Remapping Latency

- Not I/O but encode/decode
- v2: Parquet encoding cost dominates
- v3: PD files are rewritten as DV in Puffin files