Final Project Presentaion

Team 11

徐浩鈞107020022
曹竣瑋106072108
李相宇109065541

Outline

- Idea
- Implement
  - Reordering
  - Replication
- Experiments
  - Environment
  - Results
- Analyze
- Conclusion

Idea

- 針對Workload 3 : Hot Counter進行最佳化
- Hot Record作為一個時常被Read的Resource，在Transactions被分配到各個Node後，容易導致過多的Migration，降低效率
- 透過Reordering將Write的Transaction集中先行處理，再透過Replication將Record一口氣推給各個Node，達到降低Migration次數的效果

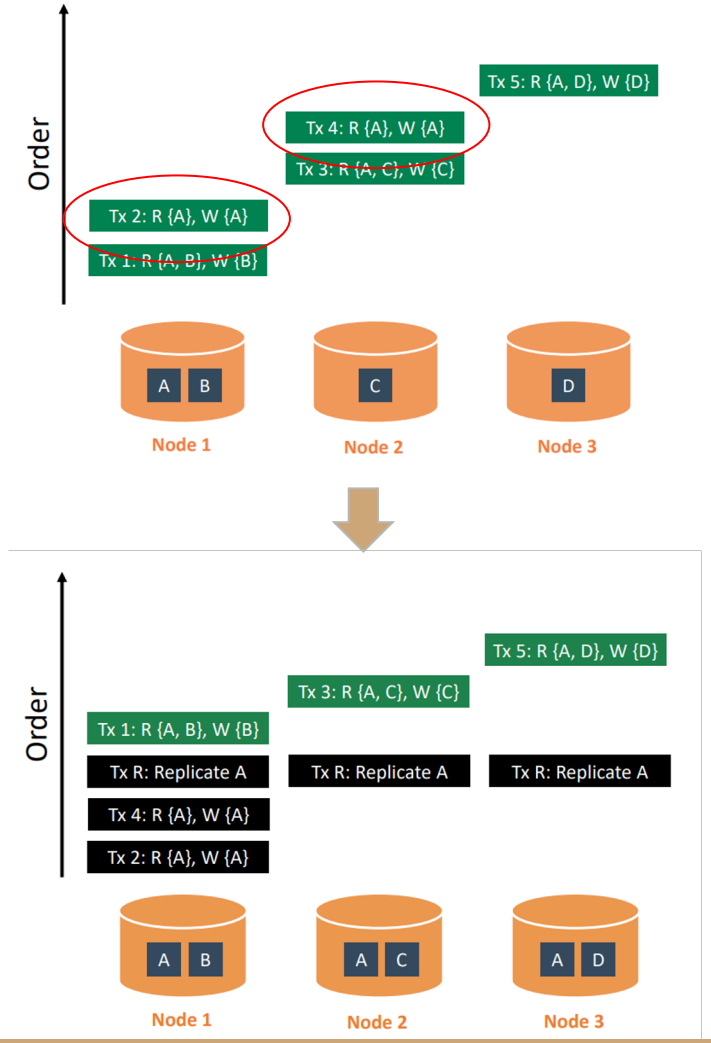
Reordering

目標:

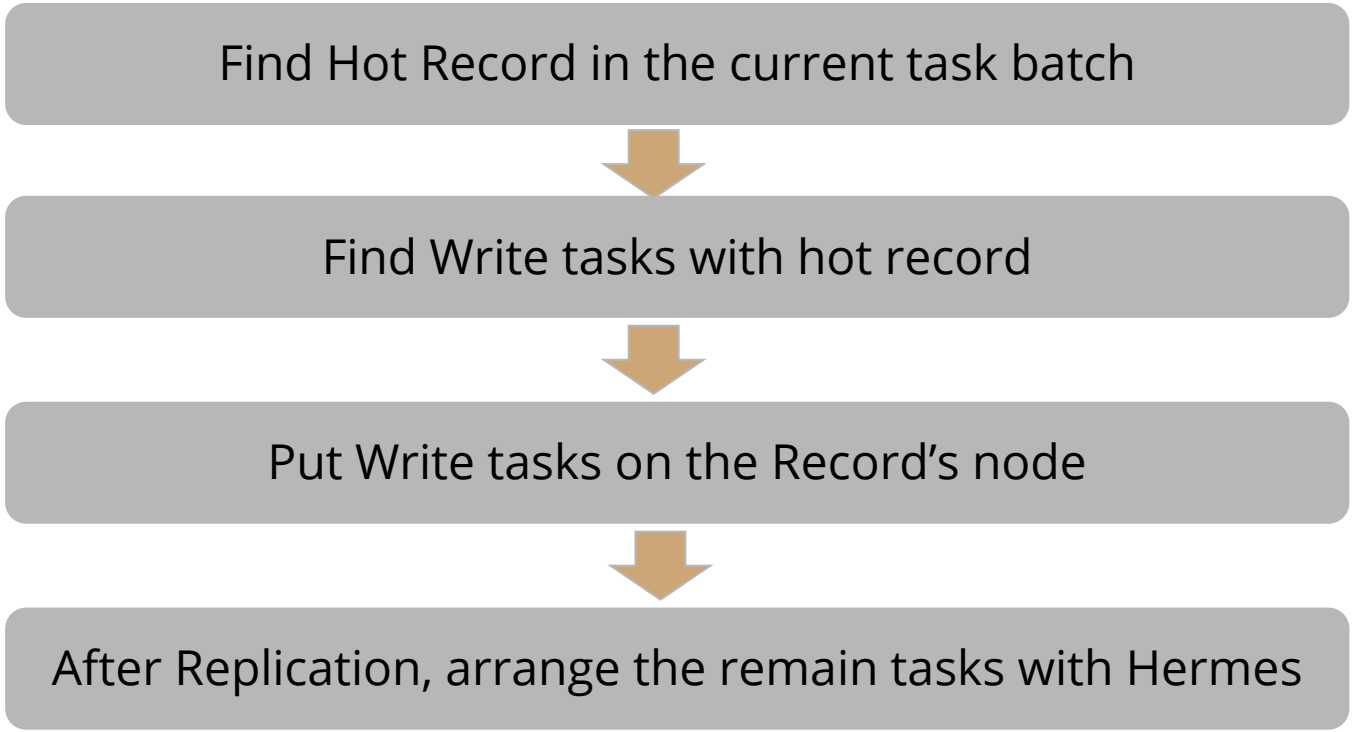
把Write Hot record的Tx擺到擁有資源的Node先行執行

把該Record複製到每個Node上

剩餘的Tx照原先Hermes的排法安排



Reordering


Find Hot Record in the current task batch
Find Write tasks with hot record
Put Write tasks on the Record’s node
After Replication, arrange the remain tasks with Hermes

Changes made for Reordering

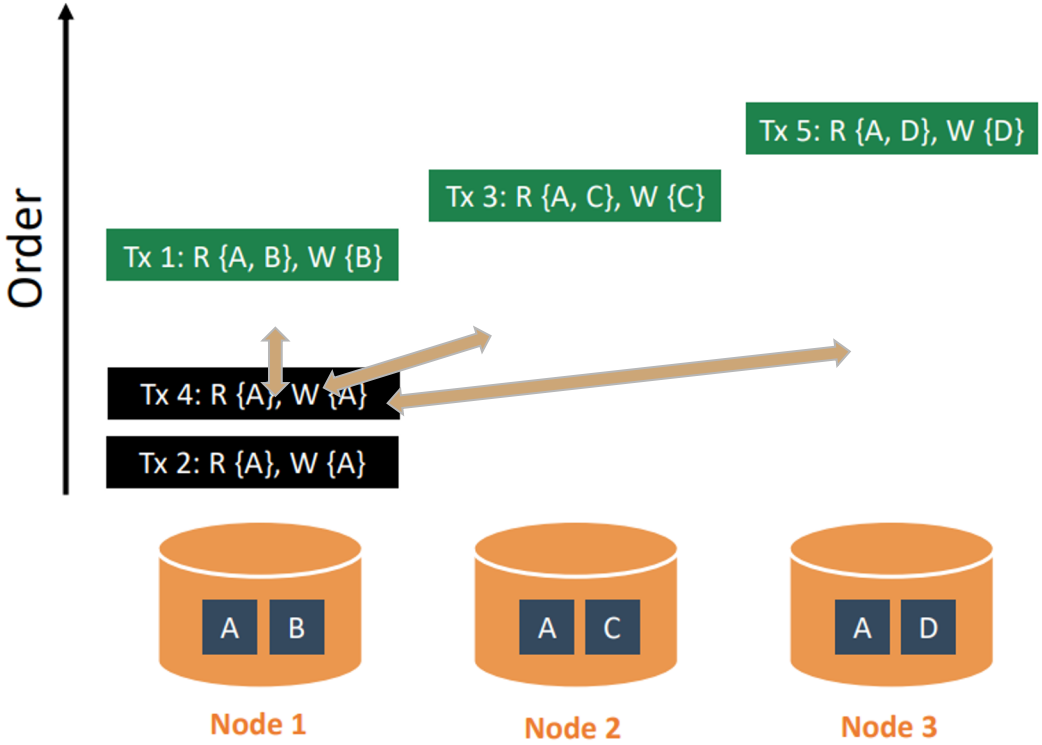
- new method insertToResourcePart()
  - Insert the task to the location of the record
- modify method processBatch()
  - Iterate through the tasklist to find the hot record
  - Iterate thorugh the tasklist to find the write task with hot record
  - Insert the write tasks but not updating the statistic of each node
  - Not changing the statistic means not affecting the original hermes
  - Insert the rest of tasks with hermes

Replication

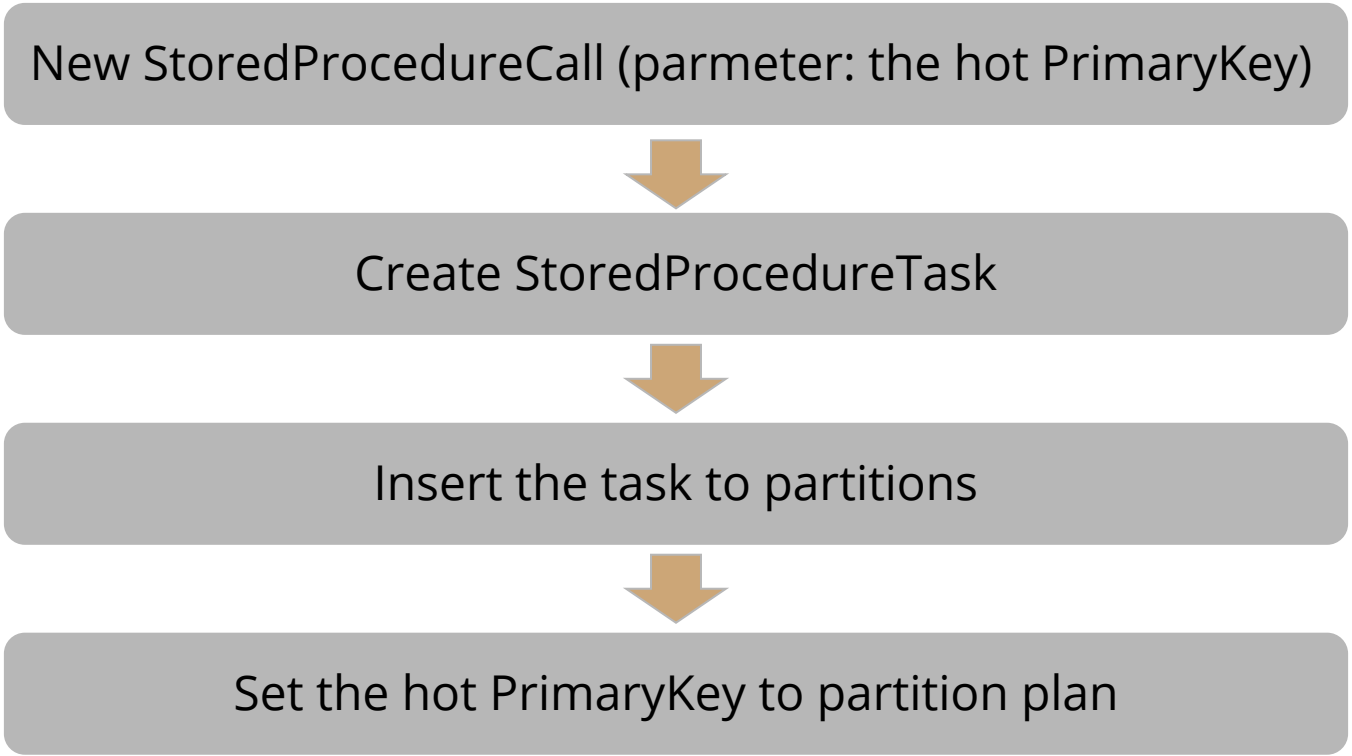
目標:

在最後write hot record的node與每個partition建立edge

紀錄hot record，讓之後Txns可以直接從cache record取用



Replication



Changes Made for Replicate

- new YcsbTransactionType : REPLICATE
  - isBenchProc = FALSE
  - add this type to TpartYcsbStoredProcFactory
- new TPartStoredProcedure : TpartReplicateProc
- new ParamHelper : ElasqlYcsbReplicateProcParamHelper
  - only one ReadKey : the hot key
- new method : setHotRecord & getHotRecord in PartitionMetaMgr
  - get & set the hot key to partition plan
- modify method : isFullyReplicated
  - true if key == the hot key

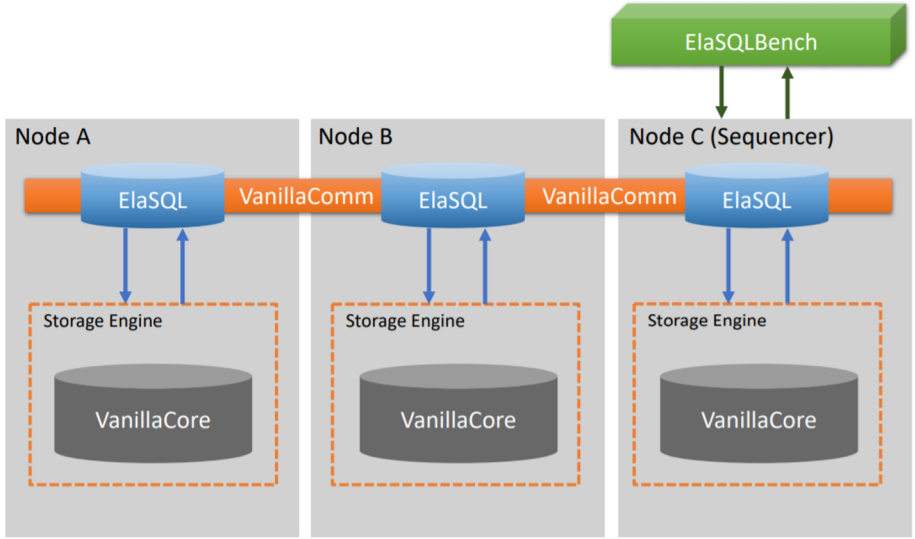
Experiments

Experiments-Enviroment

處理器	Intel(R) Core(TM) i7-10700 CPU @ 2.90GHz 2.90 GHz
已安裝記憶體(RAM)	32.0 GB (31.9 GB 可用)

處理器	Intel(R) Core(TM) i5-7200U CPU @ 2.50GHz 2.70 GHz
已安裝記憶體(RAM)	8.00 GB

處理器	Intel(R) Core(TM) i3-8100 CPU @ 3.60GHz 3.60 GHz
已安裝記憶體(RAM)	24.0 GB (23.8 GB 可用)



Experiment-Results

Workload 1 - Hotspot Workload

1. Default parameter

Hermes version

```
# of txns (including aborted) during benchmark period: 82199
YCSB - committed: 82199, aborted: 0, avg latency: 36 ms
TOTAL - committed: 82199, aborted: 0, avg latency: 36 ms
```

Improved

```
# of txns (including aborted) during benchmark period: 87770
YCSB - committed: 87770, aborted: 0, avg latency: 34 ms
TOTAL - committed: 87770, aborted: 0, avg latency: 34 ms
```

Experiment-Results

Workload 2 - Google Workload

- Default Parameter, Benchmark Interval = 120000

Hermes version

```
# of txns (including aborted) during benchmark period: 20360
YCSB - committed: 20360, aborted: 0, avg latency: 58 ms
TOTAL - committed: 20360, aborted: 0, avg latency: 59 ms
```

Improved version

```
# of txns (including aborted) during benchmark period: 21880
YCSB - committed: 21880, aborted: 0, avg latency: 54 ms
TOTAL - committed: 21880, aborted: 0, avg latency: 55 ms
```

Experiment-Results

Workload 3 - Hot Counter Workload

1. When running with default parameter, our version performs better (hot_update_rate_in_rw_tx =0.1, hot_count_per_part=1, warmup_interval = 30000)

Hermes version

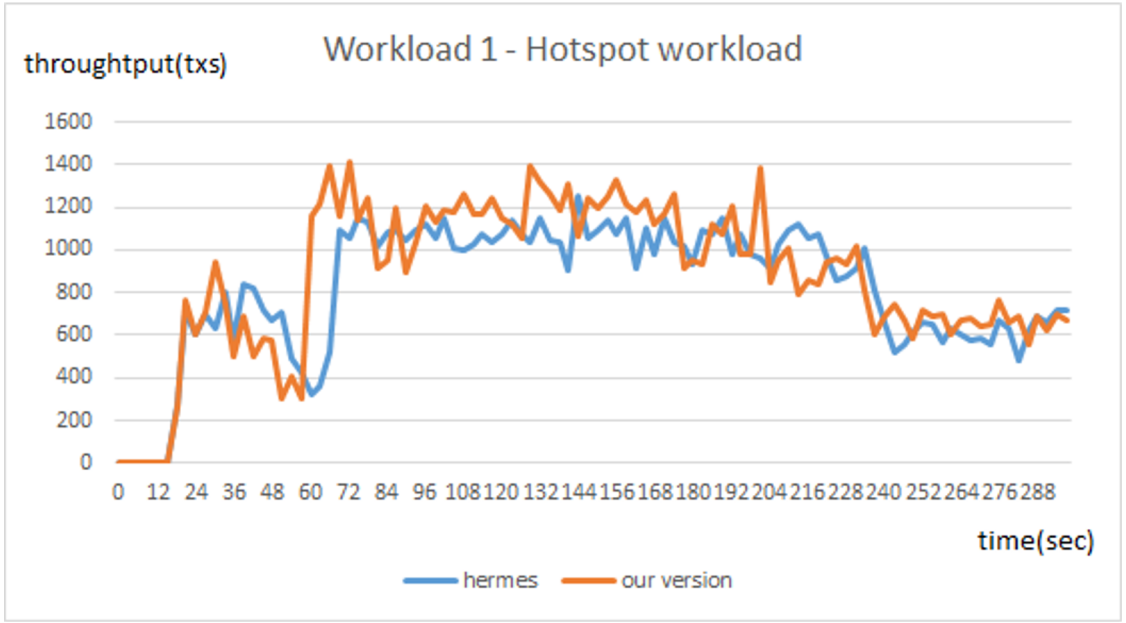
```
# of txns (including aborted) during benchmark period: 22801
YCSB - committed: 22801, aborted: 0, avg latency: 52 ms
TOTAL - committed: 22801, aborted: 0, avg latency: 53 ms
```

Improved ve

```
# of txns (including aborted) during benchmark period: 25870
YCSB - committed: 25870, aborted: 0, avg latency: 46 ms
TOTAL - committed: 25870, aborted: 0, avg latency: 46 ms
```

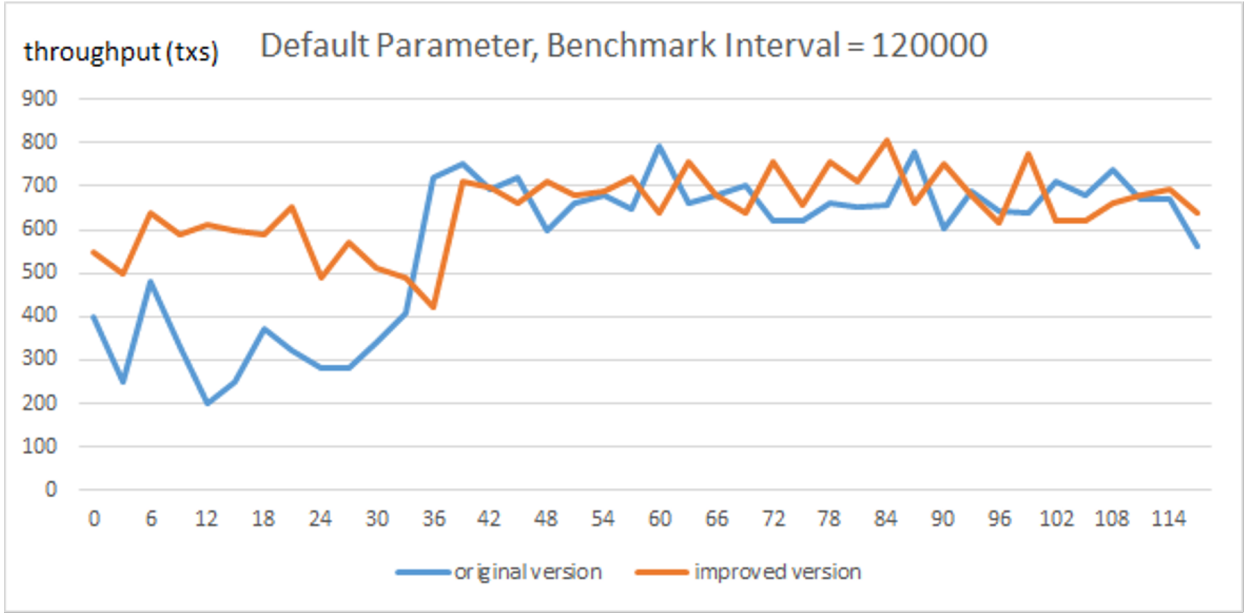
Analysis

In workload 1, why better than Hermes?



Analysis

In workload 3, performance goes well



Conclusion

- Decrease the amount of the data migrated
- Reordering and Replicate optimize workload 3
- Experiment data support our modification

Appreciate for your attention :)