2021Spring Database Final Project Report

member:
109065541李相宇
107020022徐浩鈞
106072108曹竣瑋

主要去針對Workload 3: Hot Counter進行優化。該workload主要的問題便是有部分record會被大量的transaction進行讀取，稱之為hot record，hot record在原先hermes的演算法底下會因為transaction被負載平衡後，造成大量的record migration。

我們主要使用Reordering結合Replication的技巧去解決這個問題。

Reordering:

目標:把Write hot record的transaction擺到擁有該record的partition上優先執行，並將剩餘的transaction按照Hermes去安排

實作:

步驟為

1. 找到Hot record
2. 安排Write hot record的transactions
3. 進行Replication的安排
4. 安排剩餘Transaction

修改文件TPartPartitioner.java

新增Method insertToResourcePart(TGraph graph, TPartStoredProcedureTask task, PrimaryKey resource)

:為了能將Trasaction安排到擁有資源的機器上，去對plan graph進行node的增加與edge的安排而增加的method

修改Method processBatch():

1. 為了找到Hot Record，去遍歷整個Task list，尋找一個最高read數量的record
2. 再次遍歷整個task list找出所有write task
3. 將write task透過insertToResourcePart()放入對應的機器
4. 讓剩餘的task按照原本的hermes去安排位置

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

實作：

步驟為

1. 創建新的StoredProcedureCall (parameter只有hot key)
2. 利用這個SPcall建立StoredProcedureTask
3. 將task放到每個partition上面
4. 記錄這個batch的hot key

新增新的TransectionType REPLICATE

修改文件TpartYcsbStoredProcFactory.java

新增REPLICATE 的case：可以分辨是否為REPLICATE txn

新增REPLICATE相關的class

TpartReplicateProc, ElasqlYcsbReplicateProcParamHelper:

這個porc裡面只有一個readKey: hot key，如此便可將edge接上

新增method setHotRecord & getHotRecord in PartitionMetaMgr & partitionPlan:

讓yscb partitionPlan記得我們的hot key，並修改isFullyReplicated method，當key == hot key則為true

Experiment

1. Enviroment

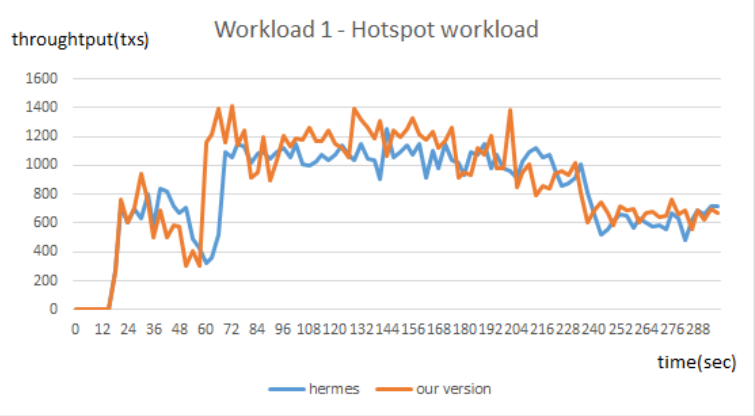
   用3台獨立機器做實驗，網路環境為同一個網路，都接網路線

   規格:

   1. Intel Core i7-10700 CPU @2.9GHz, 32GB RAM 作為Server 0
   2. Intel Core i5-7200U @2.5GHz 2.8GHz, 8GB RAM 作為Server 1
   3. Intel Core i3-8100 CPU @3.6GHz, 24GB RAM 作為Sequencer及Client

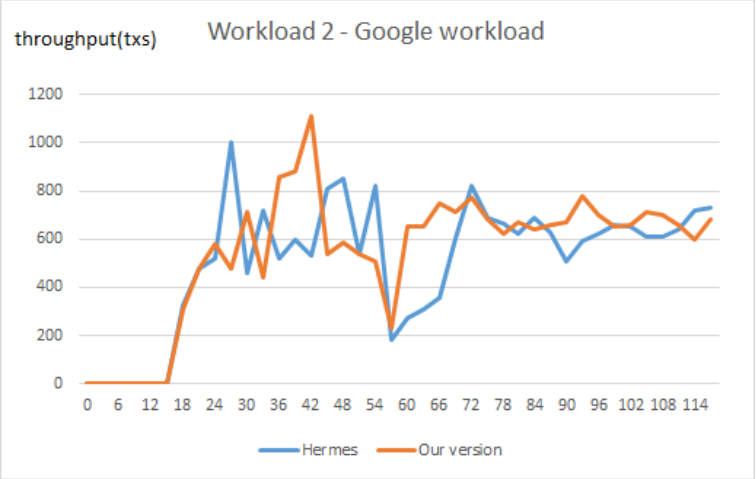
2. Data and Analysis

   1. Workload 1 - Hotspot workload



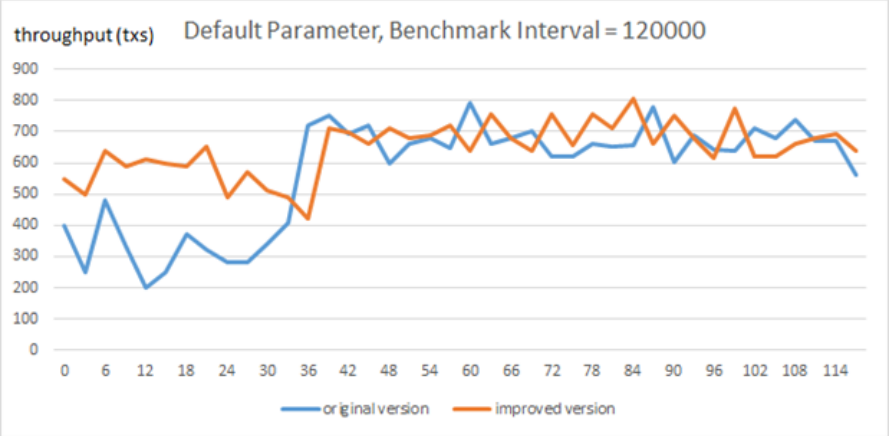
   在Workload 1中，我們版本的Transaction committed的數量比hermes稍微高，我們認為Hermes原先的作法會讓Transaction去access同一個part的record，我們replicate的部分讓一些task可以被分擔到別的part，放寬了Hermes原先的限制，稍微減輕原先那個part的負擔。

   2. Workload 2 - Google workload



Google workload中，我們的總throughput幾乎沒有比原先Hermes的版本好，加上我們認為Google workload中很難有進步的空間，也不認為我們的作法有對Google workload的部分有特別顯著的進步。

3. Workload 3 - Hot counter workload



在workload3中，這邊我們設定warmup_interval=30000，我們的作法將data migration的次數降低，將資料都先搬移到合適的地方，減少資料搬動的等待時間，因此前面的throughput會比Hermes版本的高。

Conclusion

1. 我們的作法找出hot record並且將資料先安排在合適的地方，減少Data Migration的次數，增加讀寫效率
2. 實作Replicate和Reordering的部分，最佳化workload 3

3. 實驗的數據大多support我們的做法和想法