PFLOCK Report

Andres Calderon

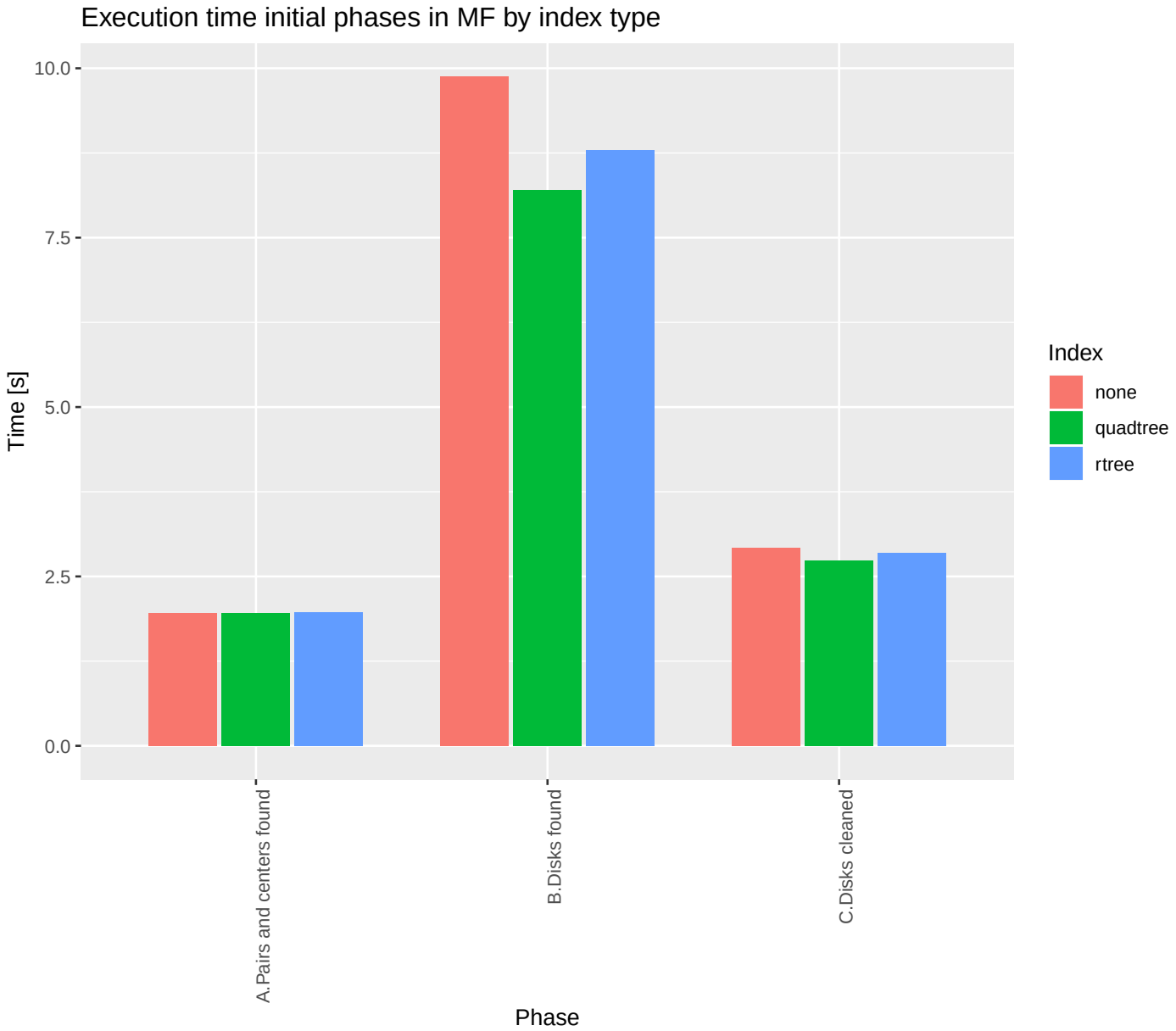
University of California, Riverside

March 6, 2020

Experiment settings...

- LA_50K dataset, time instant 320, 50419 points.
- $\mu = 3$, $\varepsilon = 45$.
- 12 executors, 9 cores each (108 cores total).
- Average time of 10 runs.
- Partitions and Parallelism set at 216.

Indexer performance...



Reading partitions before hand...

Stages for All Jobs

Completed Stages: 16
Skipped Stages: 20

Completed Stages (16)

Stage Id ▼	Description	Submitted	Duration	Tasks: Succeeded/Total	Input	Output	Shuffle Read	Shuffle Write
35	count at DJQueryFlat.scala:219 +details	2020/03/06 11:05:43	1 s	853/853			135.1 MB	
34	distinct at DJQueryFlat.scala:218 +details	2020/03/06 11:05:40	2 s	853/853			658.0 MB	135.1 MB
25	count at DJQueryFlat.scala:201 +details	2020/03/06 11:05:38	2 s	853/853			658.0 MB	
24	map at DJQueryFlat.scala:196 +details	2020/03/06 11:05:32	5 s	854/854			400.9 MB	658.0 MB
23	distinct at JoinQuery.java:508 +details	2020/03/06 11:05:19	14 s	854/854			18.1 MB	400.9 MB
22	flatMapToPair at SpatialRDD.java:334 +details	2020/03/06 11:04:46	29 s	1708/1708	45.3 MB			13.3 MB
21	flatMapToPair at SpatialRDD.java:334 +details	2020/03/06 11:04:46	4 s	853/853	4.5 MB			4.8 MB
16	aggregate at SpatialRDD.java:502 +details	2020/03/06 11:04:45	0.7 s	1708/1708	45.6 MB			
11	count at DJQueryFlat.scala:166 +details	2020/03/06 11:04:44	2 s	1708/1708	14.4 MB		34.3 MB	
10	distinct at JoinQuery.java:508 +details	2020/03/06 11:04:42	1 s	854/854	26.2 MB		3.6 MB	34.3 MB
8	flatMapToPair at SpatialRDD.java:334 +details	2020/03/06 11:04:38	4 s	853/853	4.5 MB			3.6 MB
6	count at DJQueryFlat.scala:124 +details	2020/03/06 11:04:37	1 s	854/854			5.6 MB	
5	flatMapToPair at SpatialRDD.java:334 +details	2020/03/06 11:04:33	4 s	853/853	4.6 MB			5.6 MB
3	aggregate at SpatialRDD.java:502 +details	2020/03/06 11:04:32	0.6 s	853/853	5.0 MB			
1	count at DJQueryFlat.scala:88 +details	2020/03/06 11:04:30	2 s	853/853			1204.6 KB	
0	rdd at DJQueryFlat.scala:78 +details	2020/03/06 11:04:26	4 s	1/1	1721.7 KB			1204.6 KB

Reading partitions before hand...

	Submitted	Duration	Tasks: Succeeded/Total	Input	Output	Shuffle R
+details	2020/03/06 11:05:43	1 s	853/853			135.1 MB
+details	2020/03/06 11:05:40	2 s	853/853			658.0 MB
+details	2020/03/06 11:05:38	2 s	853/853			658.0 MB
+details	2020/03/06 11:05:32	5 s	854/854			400.9 MB
+details	2020/03/06 11:05:19	14 s	854/854			18.1 MB
+details	2020/03/06 11:04:46	29 s	1708/1708	45.3 MB		
+details	2020/03/06 11:04:46	4 s	853/853	4.5 MB		
+details	2020/03/06 11:04:45	0.7 s	1708/1708	45.6 MB		
+details	2020/03/06 11:04:44	2 s	1708/1708	14.4 MB		34.3 MB
+details	2020/03/06 11:04:42	1 s	854/854	26.2 MB		3.6 MB
+details	2020/03/06 11:04:38	4 s	853/853	4.5 MB		
+details	2020/03/06 11:04:37	1 s	854/854			5.6 MB
+details	2020/03/06 11:04:33	4 s	853/853	4.6 MB		
+details	2020/03/06 11:04:32	0.6 s	853/853	5.0 MB		
+details	2020/03/06 11:04:30	2 s	853/853			1204.6 KI
+details	2020/03/06 11:04:26	4 s	1/1	1721.7 KB		

Reading partitions before hand...

Some current issues...

- The parallelism parameter cannot be updated in Runtime.
- The previous figures extends the default GeoSpark partitioner to read a predefined set of cells (in this case a quadtree) but performance gain is lost.
- Using a custom quadtree loses integration with the available GeoSpark operations.
- Currently I have move the code of the GeoSpark's quadtree to Scala and hack it a bit to be able to read cells from disk (still working on it).