SIGMOD RWE Review
"Efficient Parallel Set-Similarity Joins Using MapReduce"

Fabian Hueske, TU Berlin

June 3, 2010

1 Setup

This document is a review report on the paper "Efficient Parallel Set-Similarity Joins Using MapReduce" by R. Vernica, M. Carey, C. Li by Sigmod's 2010 Repeatability and Workability Evaluation Committee.
In this section the provided resources (code, data sets, setup information) and hardware setups of the authors and reviewers are discussed. Detailed information on all experiments that the reviewer conducted or tried to conduct for repeatability or workability can be found in sections 2 and 3.

1.1 Provided Resources

The resources provided by the authors were very well prepared. Documentation contained the relevant information to conduct all experiments. The authors experiments were assembled from a set of several bash and perl scripts, that orchestrated the full pipeline of the experiments, i.e. formating the distributed file system, uploading original data, generating test data, running experiments, plotting figures.
Some scripts required minor adaptations to fit the reviewer's setup. Due to the documentation and self-explaining nature of the scripts, that was an easy task.
The full script automation of the experiments tremendously eased the repetition of the experiments.

1.2 Comparing Experimental Setups

The experiments of the paper were run on cluster consisting of 10 IBM x3650 worker machines and a coordinator machine. The worker machines had one Intel Xeon QuadCore E5520 CPU running at 2.26GHz, 12GB RAM and four 300GB hard disks which did not run in RAID mode. The coordinator machine was not specified in detail. All machines ran Ubuntu Linux 9.04 (64bit, server edition), Java 1.6 (64bit, server JVM), and Hadoop 0.20.1. The authors configured Hadoop to run 4 map and 4 reduce tasks on each worker. The JVMs running those tasks had at maximum 2,5GB RAM available. Repeated execution of failed tasks and speculative execution were turned off. The data of a HDFS datanode was spread over all four disks.
The reviewer's experiments were conducted on a cluster consisting of 5 servers. Each server was equipped with two AMD Opteron QuadCore CPUs running at 2.3GHz, 32GB RAM, and four 146GB hard disks (one system disk, three data disks bundled with RAID0). The machines ran Ubuntu Linux 8.04 LTS (64bit, server edition, updates from 05/27/2010), Java 1.6, and Hadoop 0.20.2. The reviewer aimed to configure Hadoop the same way the authors did. However, following modifications to the original setup were necessary:

- Jobtracker and Namenode did not run on a dedicated machine for the experiments conducted with all 5 machines. Experiments with less than 5 machines had a dedicated Jobtracker and Namenode.
- A HDFS datanode's data was stored on the 3-disk RAID0 device.
- Hadoop's feature to restart failed tasks was enabled. This was necessary due to occasional not reproducible task failures which caused the failure of a whole experiment. Exceptions stated that a local directory was not writable. The reviewer suspects the local file system of being responsible for those exceptions.

Although the reviewer's machines seem to be approximately twice as powerful than the authors machines (number of cores, amount of RAM), the reviewer was not able to run twice as many map and reduce jobs on each node. Such attempts failed with exceptions stating lack of available memory. Hence, the reviewer chose to run the experiments with identical configuration as the authors with respect to number of map and reduce tasks and amount of RAM given to task JVMs.

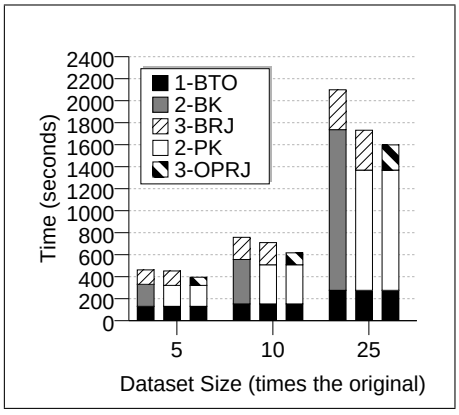

Figure 1: Repeated Fig 8

1.3 Interaction with Authors

The authors were very helpful and responded fast to the reviewers questions. The reviewer had some issues with the setup of the Hadoop system. The author's gave useful advices to solve the problem (e.g., enabling repeated execution of failed tasks, which solved one main issue with the Hadoop setup). The reviewer was able to conduct all experiments without interaction with the authors, due to the complete documentation and excellent setup of scripts provided by the authors.

2 Repeatability Evaluation

In this section detailed information on the experiments to repeat the result presented in the paper is given. For each experiment, the reviewer adapted the provided run script to the hardware setup and executed it. After completion, the figures and runtime data were included into this report.

2.1 Figure 8 - Performance DBLP

Figure 1 shows the result of the experiments to repeat Figure 8 from the original paper. The reviewer ran the experiment one 5 machines while the authors used 10. Besides the absolute runtimes, both figures are almost identical.

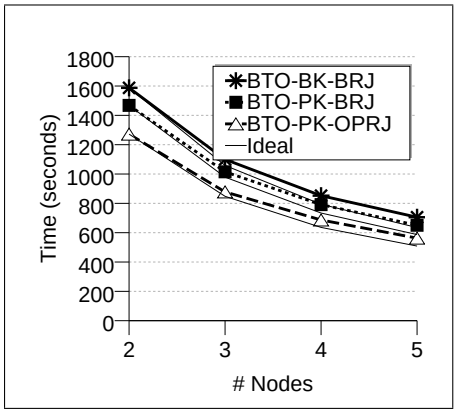

Figure 2: Repeated Fig 9

2.2 Figure 9, 10 & Table 1 - Speedup DBLP

Figure 2 was derived from experiments repeating Figure 9 of the paper. The original experiments were ran on varying cluster sizes (2,4,8,10 nodes) while the reviewer used 2, 3, 4, and 5 nodes to repeat the results. Compared to the original figure, the repeated Figure shows an increased runtime of 25% to 30%. However, the trend of all curves is very similar. The increase of runtime can be attributed to the difference in the HDFS setup. In the authors setup, all map tasks have exclusive access to a single disk, while in the reviewers setup 4 map tasks compete for a 3-disk RAID0.

Figure 2.2 repeats Figure 10 from the original paper. The figures were generated from the same runtime data as Figure 2 / Figure 9. Both figures indicate a very similar speed-up behavior.

Table 1 shows the absolute values for the runtimes of the individual algorithms that were observed from the experiments to derive figure 2 and 2.2. Hence it repeats Table 1 from the original paper. In general, we observe longer runtimes compared to the original experiments. Both tables show very similar relative differences for competing algorithms and varying number of nodes. Looking closer at the algorithms for the first stage, the original paper indicates a better scaling of the BTO algorithm (starts to outperform OPTO somewhere between 5 and 8 nodes. In the repeated experiments, BTO already betters OPTO on a 4-node setup.

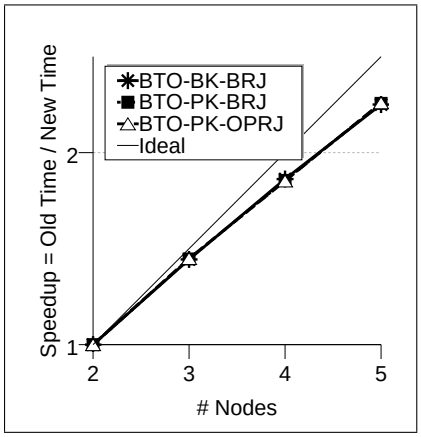

Figure 3: Repeated Fig 10

Stage	Alg.	No. Nodes			
		2	3	4	5
1	BTO	254.15	196.12	160.78	144.70
	OPTO	241.54	186.18	165.42	150.26
2	BK	1018.02	668.29	501.53	389.17
	PK	898.51	583.78	440.40	334.33
3	BRJ	315.35	235.14	190.09	171.52
	OPRJ	119.34	98.08	85.43	83.91

Table 1: Repeated Table 1

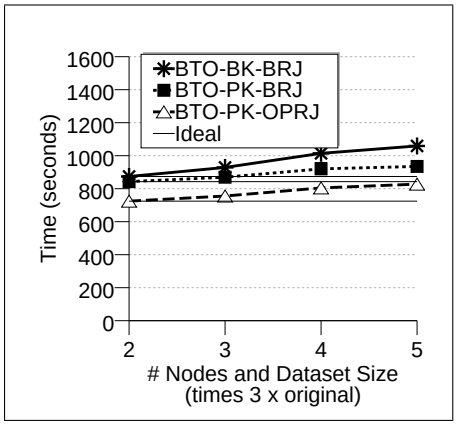

Figure 4: Repeated Fig 11

2.3 Figure 11 & Table 2 - Scaleup DBLP

Figure 4 was derived by repeating the experiments for Figure 11 of the original paper. The setup was adapted to fit the reviewers test environment. While the authors ran experiments for 2, 4, 8 and 10 nodes, the reviewer repeated the experiment with 2, 3, 4, and 5 machines. Consequently, the scale factor of the data set was adapted. The reviewer chose an $no.nodes/scalefactor$ ratio of 3 compared to 2.5 in the original setting to avoid complications caused by factional scale factors for uneven numbers of nodes (sf 7.5 for 3, and 12.5 for 5 nodes).
Comparing Figure 4 and Figure 11 of the original paper, we observe longer runtimes (different HDFS setup and higher $no.nodes/scalefactor$ ratio). Both figures indicate a similar scale-up behavior of the proposed methods.
Table 2 gives the runtimes of the different algorithms that were observed during the experiment to derive Figure 4. It corresponds to Table 2 in the original paper. Of course, absolute numbers vary but the the fundamental trends such as better scale-up behavior of BRJ compared to OPRJ and BTO compared to OPTO could be repeated.

2.4 Figure 12 - Performance DBLP+CiteseerX

Figure 5 shows the results of the repeated experiments for Figure 12 from the original paper. For this experiment, the reviewer used 5 machines, while

Stage	Alg.	No. Nodes / Dataset Size			
		2/x6	3/x9	4/x12	5/x15
1	BTO	172.11	171.12	190.19	174.07
	OPTO	158.38	173.40	182.13	194.36
2	BK	509.52	560.60	619.59	672.64
	PK	478.37	501.47	525.51	548.54
3	BRJ	192.21	197.05	204.45	212.29
	OPRJ	73.96	83.02	88.96	105.05

Table 2: Repeated Table 2

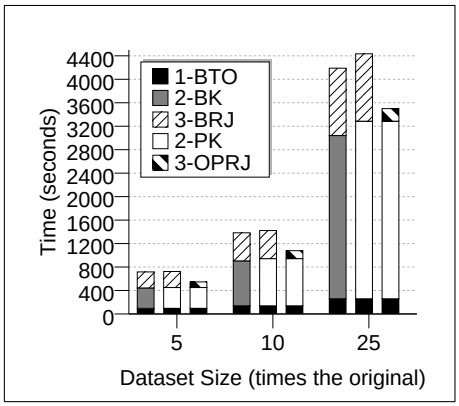

Figure 5: Repeated Fig 12

the authors used 10. Comparing both figures, two observations can be made. First, in the reviewers results, the OPRJ phase requires significantly less time, compared to the BRJ algorithm. Second, the reviewer was able to successfully run OPRJ even with scale factor 25 which failed in the authors setup due to lack of available memory. The fact that the reviewer's machines had more RAM (32GB compared to 12GB) is a possibly explanation for both observations. The increased runtime might be attributed to the usage of swap space in the authors setup.

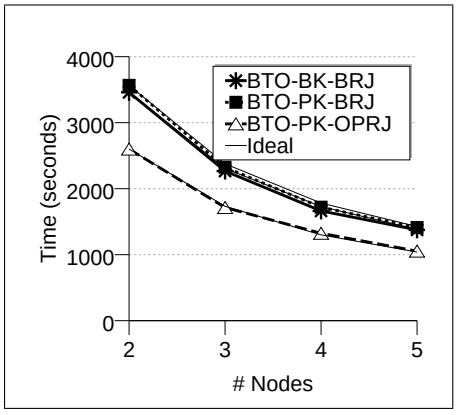

Figure 6: Repeated Fig 13

2.5 Figure 13 - Speedup DBLP+CiteseerX

Figure 6 repeats Figure 13 of the original paper. Similar as for Figure 2 the reviewer ran the experiments on 2 to 5 nodes. The authors used 2, 4, 8, and 10 nodes. Given the observation from Figure 5, the better performance of the test runs including the OPRJ algorithm on the reviewers setup can be explained with the larger amount of available memory. Apart from that, both figures indicate similar speed-up behaviors.

2.6 Figure 14 - Scaleup DBLP+CiteseerX

Figure 7 shows the results of repeating the experiments for Figure 14 of the original paper. The reviewer used the same $no.nodes/scalefactor$ ratio as for Figure 4. The reviewer's results show a better scaleup behavior for the test runs that include the OPRJ algorithm, which again can be explained with the better equipped hardware used in the reviewer's setup. Another difference is the the slightly better performance of the BTO-BK-BRJ compared to BTO-PK-BRJ in the reviewers results while Figure 14 of the authors show that BTO-PK-BRJ outperforms BTO-BK-BRJ. However, both methods show similar scale-up behavior in both figures.

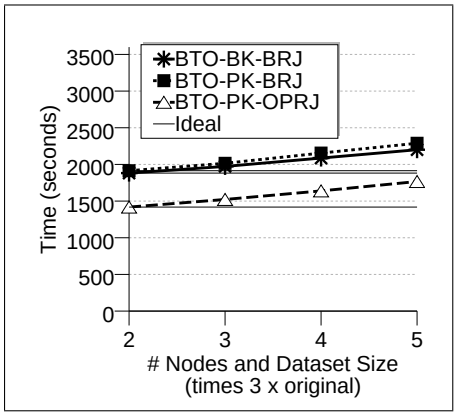

Figure 7: Repeated Fig 14

3 Workability Evaluation

Workability tests were not performed, due limited time.

4 Conclusion

The reviewer was able to run all experiments on a half-sized hardware setup compared to the author's setup. All experimental results have been fully reproduced. Due to the good documentation and excellently scripted experimental setup, the reviewer's actions were limited to setup Hadoop, start the experiments, wait, and collect and inspect the results.
Since, most experiments ran for more than 12 hours and initial problems with the Hadoop setup, the reviewer was not able to conduct a workability evaluation.