BAB 6 PENUTUP

1.1 Kesimpulan

Berdasarkan dari hasil proses implementasi, pengujian, dan analisis yang telah dilakukan untuk mengetahui kinerja algoritme *Fair Share Scheduling* dan *Capacity Scheduling* terhadap pengiriman *job* pada Hadoop, maka dapat ditarik kesimpulan sebagai berikut :

1. Algoritme *Fair Share Scheduling* dan *Capacity Scheduling* dirancang dan disimulasikan menggunakan *framework* Hadoop dengan versi 2.7.3. Pada Hadoop ini menggunakan *cluster multinode* dengan *node* berjumlah 6 buah, 1 sebagai komputer *master* dan 5 sebagai komputer *slave*. Pada Hadoop multi*node* ini dapat melakukan pengiriman *job* dari *slave* ke *master* Hadoop berdasarkan skenario pengujian yang ditentukan. Pengujian sistem yang dilakukan pada penelitian ini dilakukan dengan tiga skenario yang berbeda berdasarkan pada variabel bebas dan variabel tetap. Untuk skenario pengujian pertama menggunakan variabel bebas berdasarkan variasi ukuran data, skenario pengujian kedua menggunakan variabel bebas berdasarkan variasi jumlah *job*, dan skenario pengujian ketiga menggunakan variabel bebas berdasarkan variasi jenis *job*.
2. Pada saat pengujian dengan membandingkan algoritme *Fair Share Scheduling* dan *Capacity Scheduling*, hasil yang diperoleh berdasarkan parameter pengujian yaitu, Nilai dari parameter *Job Fail Rate* pada *Capacity Scheduling* lebih baik jika dibandingkan dengan *Fair Share Scheduling* dengan nilai minimal sebesar 0.99% sedangkan pada *Fair Share Scheduling* nilai minimal sebesar 1.32% pada scenario dengan variasi ukuran data. Karena *Capacity Scheduling* semakin bertambahnya *job* yang dijalankan maka semakin memperkecil nilai *timeout* sehingga dapat mengalami penurunan pada nilai failed rate. Nilai dari parameter *Latency* pada *Fair Share Scheduling* lebih baik jika dibandingkan dengan *Capacity Scheduling* dengan nilai minimal sebesar 19,25% sedangkan pada *Capacity Scheduling* nilai minimal sebesar 39.35% pada scenario dengan variasi ukuran data. Karena pada *Fair Share Scheduling* dapat melakukan r*unning job* sebanyak 3 *job* pada waktu yang bersamaan dengan membagi *resource* secara adil antara *job* satu dengan *job* lainnya, sehingga proses pengerjaan *job* akan lebih cepat selesai jika dibandingkan dengan *Capacity Scheduling.* Nilai dari parameter *Throughput* pada *Fair Share Scheduling* memiliki kecepatan mengirim data yang lebih baik jika dibandingkan dengan *Capacity Scheduling* dengan nilai maksimal sebesar 0,47Mbit/s sedangkan pada *Capacity Scheduling* nilai minimal sebesar 0,36 Mbit/s. Pada setiap melakukan pengiriman *job* pada suatu antrian terjadi *timeout* yang dapat menghambat kecepatan mengirim data menjadi tidak efektif dalam menentukan kinerja algoritme scheduling.

1.2 Saran

Berdasarkan hasil pengujian dan analisis yang telah dilakukan penulis, pada penelitian ini dapat dilakukan pengembangan lebih lanjut. Ada beberapa saran untuk para peneliti yang ingin melakukan pengembangan pada penelitian ini sebagai berikut.

1. Melakukan modifikasi pada jenis *job* yang berbeda dengan penelitian ini.
2. Menambahkan jumlah *job* sesuai kebutuhan.
3. Melakukan modifikasi pada parameter pengujian yang berbeda pada penelitian ini.
4. Mengganti algoritme *job* scheduling Hadoop lain pada s*erver* Hadoop.