BAB 6

KESIMPULAN DAN SARAN

Pada bab ini dibahas mengenai kesimpulan dan saran berdasarkan semua penelitian, studi eksplorasi, dan eksperimen yang telah dilakukan.

6.1 Kesimpulan

Berdasarkan studi literatur dan eksperimen yang sudah dilakukan, berikut ini adalah kesimpulan yang berhasil ditarik.

1. Hadoop dan Spark digunakan untuk mengolah data pada sistem yang terdistribusi, yaitu sebuah sistem yang terdiri atas klaster mesin-mesin, tetapi kedua *framework* tersebut memiliki perbedaan pada cara pengolahan data.

2. Pengolahan data pada Hadoop dilakukan dengan menggunakan MapReduce, yaitu dengan melakukan proses pemetaan pasangan *key* dan *value* menjadi *list key* dan *value* yang kemudian direduksi menjadi pasangan *key* dan *value*. Pada setiap tahapan pemrosesan tersebut, terjadi proses membaca dan menulis data secara langsung pada *hard drive*.

3. Pada Spark, data yang diolah dimasukkan ke dalam sebuah RDD dan dioperasikan dengan menggunakan operasi-operasi transformasi dan aksi. Operasi transformasi bersifat *lazy*, sehingga tidak akan dijalankan sebelum ada operasi aksi yang dilakukan. Sifat operasi yang *lazy* dan fakta bahwa proses membaca atau menulis data pada *hard drive* terjadi hanya pada awal dan akhir pemrosesan membuat kecepatan pemrosesan menggunakan Spark menjadi lebih cepat.

4. Pengolahan graf dengan GraphX dilakukan dengan melakukan operasi pada dua buah RDD pembentuk graf, yaitu RDD untuk simpul dan RDD untuk sisi. Akan tetapi, pembuatan graf tersebut dapat dilakukan hanya dengan menggunakan RDD sisi saja.

5. Implementasi perangkat lunak demo dilakukan dengan menggunakan bahasa pemrograman Scala dan memanggil perintah-perintah Spark untuk melakukan praolah data sebelum memanggil perintah-perintah GraphX untuk melakukan analisis. Hasil analisis dan waktu eksekusi untuk setiap perintah disimpan di dalam HDFS dan ditampilkan oleh perangkat lunak web yang diimplementasikan dengan menggunakan bahasa pemrograman Java.

6. Waktu eksekusi tidak sepenuhnya tergantung pada ukuran data, tetapi juga tergantung pada cara kerja masing-masing algoritma tersebut.

7. Ukuran data memiliki pengaruh cukup besar untuk waktu eksekusi *page rank* dan *triangle count* dibandingkan dengan algoritma-algoritma lain karena sifat kedua algoritma tersebut yang menelusuri keseluruhan graf yang terdistribusi.

8. Waktu eksekusi untuk *connected component* tidak terlalu terpengaruh oleh ukuran data karena algoritma tersebut berjalan dengan memeriksa seluruh simpul pada graf secara paralel.

9. Waktu eksekusi untuk *shortest paths* lebih banyak dipengaruhi oleh simpul-simpul sumber dan tujuan serta hasil partisi graf dibandingkan dengan ukuran data karena algoritma tersebut mencari jalur di antara kedua simpul yang dapat terletak pada partisi berbeda.
10. Waktu eksekusi untuk *subgraph* merupakan yang terlama dibandingkan dengan algoritma lain karena adanya proses pengubahan graf tersebut menjadi format JSON.
11. Waktu eksekusi algoritma cenderung lebih tinggi ketika menggunakan set data berukuran kurang dari 1.2 GB. Dengan demikian, batasan ukuran data yang cocok untuk diolah menggunakan Spark adalah 1.2 GB untuk ukuran klaster empat node dengan spesifikasi yang telah dijabarkan pada 5.1.1 dan 5.1.2.
12. Waktu eksekusi tergantung pada beberapa faktor, yaitu ukuran data, jenis algoritma, hasil partisi graf yang dihasilkan, dan masukan algoritma.

6.2 Saran

Berdasarkan kesimpulan yang telah dijabarkan, berikut ini adalah saran-saran yang dapat diajukan.

1. Untuk meningkatkan kecepatan pemrosesan, diperlukan tambahan mesin ke dalam klaster. Semakin banyak mesin yang ada dalam klaster, maka data yang besar dapat diolah dengan lebih cepat.
2. Karena pemrosesan Spark berjalan di dalam memori, maka setiap mesin dalam klaster sebaiknya mempunyai ukuran memori yang cukup besar.
3. Kualitas jaringan yang baik akan mendukung koneksi antarmesin klaster selama proses berjalan.

DAFTAR REFERENSI

[1] Guller, M. (2015) *Big Data Analytics with Spark: A Practitioner's Guide to Using Spark for Large Scale Data Analysis*, 1st edition. Apress, New York City.

[2] Upadhyaya, B. P. (2017) *Programming with Scala. Language Exploration.* Springer, Berlin.

[3] Michael Malak, R. E. (2016) *Spark GraphX in Action.* Manning Publications, Greenwich.

[4] Deo, N. (2016) *Graph Theory with Applications to Engineering and Computer Science*, reprint edition Dover Books on Mathematics. Dover Publications, Mineola.

[5] Thomas Erl, P. B., with Wajid Khattak (2016) *Big Data Fundamentals: Concepts, Drivers & Techniques* The Prentice Hall Service Technology Series from Thomas Erl. Prentice Hall, Upper Saddle River.

[6] Holmes, A. (2012) *Hadoop in Practice.* Manning Publications, Greenwich.

[7] Karau, H., Andy Konwinski, P. W., dan Zaharia, M. (2015) *Learning Spark: Lightning-Fast Big Data Analysis*, 1st edition. O'Reilly Media, Sebastopol.

[8] Team, O. R. (2012) *Big data now: 2012 edition*, 2nd edition. O'Reilly Media, Sebastopol.

[9] Rosen, K. H. (2011) *Discrete Mathematics and Its Applications*, 7th edition. McGraw Hill Higher Education, New York City.

[10] Roughgarden, T. (2014) Counting triangles. *Reading in Algorithms*, **1**.

[11] Lam, C. (2010) *Hadoop in Action*, 1 edition. Manning Publications, Greenwich.

[12] Scott, J. A. (2015) *Getting Started with Apache Spark*, 1st edition. MapR Technologies, San Jose.

[13] Ryza, S., Uri Laserson, S. O., dan Wills, J. (2015) *Advanced Analytics with Spark: Patterns for Learning from Data at Scale*, 1st edition. O'Reilly Media, Sebastopol.

[14] Xin, R. S., Gonzalez, J. E., Franklin, M. J., dan Stoica, I. (2013) Graphx: A resilient distributed graph system on spark. *First International Workshop on Graph Data Management Experiences and Systems*, **1**.