최신 분산 그래프 처리 시스템에서의 PageRank/BFS 질의 처리 성능 평가

이경준*, 김현지*, 이유경*, 이준영**†, 김강수*, 한욱신***
*포항공과대학교 창의 IT 융합 공학과
**포항공과대학교 컴퓨터 공학과
***포항공과대학교 창의 IT 융합공학과/컴퓨터공학과
e-mail : kjlee@dblab.postech.ac.kr, hjkim@dblab.postech.ac.kr, yklee@dblab.postech.ac.kr, juneyoung.lee@sf.snu.ac.kr, kskim0610@postech.ac.kr, wshan@postech.ac.kr

Experimental Evaluation of PageRank/BFS Queries on Distributed Graph Processing Systems

Kyeong-Jun Lee*, Hyeonji Kim*, Yukyoung Lee*, Juneyoung Lee**, Kangsu Kim*, Wook-Shin Han***
*Dept. of Creative IT Convergence Engineering, POSTECH
**Dept. of Computer Science, POSTECH
***Dept. of Creative IT Engineering/Dept. of Computer Science and Engineering, POSTECH

요 약

그래프는 객체와 객체 간의 관계를 표현하는 데에 있어 효과적인 데이터 표현 방법이다. 그래프 데이터는 웹 그래프, 사회 관계망 서비스, 신약 개발, 생명정보학 등의 다양한 분야에서 활용되고 있으며, 그래프 마이닝 응용에서 활용되기 위해 효율적인 처리 기술을 필요로 한다. 최근까지 그래프 데이터의 처리 및 분석을 위한 많은 시스템들이 개발되었다. 본 논문에서는 최신 분산 그래프 처리 시스템 중에서 대표적인 그래프 분석 질의인 페이지랭크(pagerank)와 너비 우선 탐색(breadth first search) 를 수행하고 시스템의 성능을 평가한다.

1. 서론

그래프는 현실 세계에 존재하는 객체 간의 관계를 표현하는 효과적인 데이터 표현 방법이다. 이러한 그래프는 일반적으로 객체를 표현하는 정점과 정점 간의 관계를 나타내는 간선으로 이루어진다. 최근 그래프 데이터는 웹 그래프, 사회 관계망 서비스, 신약 개발, 생명정보학 등의 다양한 분야에서 활용되고 있다. 그래프 데이터의 크기가 증가하고, 그래프 마이닝 응용의 필요성이 대두됨에 따라 많은 그래프 처리 시스템들이 제안 되었다 [1-6]. 본 논문에서는 분산 그래프 처리 시스템에서 대표적인 그래프 분석 질의인 페이지랭크와 너비 우선 탐색을 수행하여 시스템의 성능을 평가한다.

그래프 데이터의 크기가 증가하여 메모리 상에 전체 그래프 데이터를 상주시키기가 어려워지면서, 디스크 기반의 분산 그래프 처리 시스템들이 연구되었다[4, 7]. 디스크 기반의 분산 그래프 처리 시스템은 그래프 데이터를 디스크에서 메모리로 로드 할 때 막대한 양의 디스크 입출력을 요구하며, 다른 머신에 저장된 데이터를 가져오는 과정에서 네트워크 통신상의 병목 현상이 발생하는 등의 문제점이 존재하고 있다. 본 논문에서는 디스크 기반의 최신 분산 그래프 처리 시스템인 GraphD [6]와 Chaos [4]를 분석하고 성능을 비교하였다.

본 논문은 다음과 같이 구성된다. 제 2 장에서는 그래프 기반의 분산 시스템인 GraphD 와 Chaos 에 대해서 설명한다. 제 3 장에서는 본 논문의 실험에서 사용하는 두 가지 질의인 페이지랭크와 너비 우선 탐색의 특성에 대해서 설명한다. 마지막으로 제 4 장에서는 실험을 통해 두 시스템의 성능을 비교하고 분석한다.

2. 그래프 처리 시스템

대용량 그래프를 효율적으로 처리하기 위해 다양한 그래프 처리 시스템들이 제안되었다. 그래프 처리 시스템은 연산을 담당하는 컴퓨터 수에 따라 단일 머신 그래프 처리 시스템과 분산 그래프 처리 시스템으로 나뉜다. 단일 머신 그래프 처리 시스템은 대용량 그래프 데이터를 저장할 공간과 질의 수행을 위한 연산 능력이 부족하다는 한계점을 가지고 있다. 이러한 한계를 극복하기 위해 GraphD [6], Chaos [4], PowerGraph

"이 논문은 삼성전자 미래기술육성센터의 지원을 받아 수행된 연구임" (과제 번호 SRFC-IT1401-04)
†이 연구원은 포항공과대학교 컴퓨터공학과 소속으로 이 연구에 참여하였으며, 현재는 서울대학교 컴퓨터공학부 소속이다.

[1], Pregel+ [5] 등 과 같은 분산 그래프 처리 시스템이 제안되었다.

한편, 그래프 처리 시스템은 전체 그래프 데이터가 저장되는 위치에 따라 메모리 내(in-memory) 그래프 처리 시스템과 디스크 기반(disk-based) 그래프 처리 시스템으로 나뉜다. 메모리 내 그래프 처리 시스템은 전체 그래프 데이터를 메모리 내에 상주시킨 후 질의를 처리하므로 질의 수행 시간은 짧지만, 대용량의 저장 공간을 확보하기 위해서 디스크에 비해 상대적으로 비싼 메모리를 구매해야 하는 비용 문제가 존재한다.

본 논문에서는 디스크 기반의 분산 그래프 처리 시스템에 초점을 맞추어 대표적인 GraphD 와 Chaos 를 분석한다.

2.1. GraphD

GraphD [6]는 하드웨어 자원을 효율적으로 활용하는데 초점을 맞추어 설계된 디스크 기반의 분산 그래프 처리 시스템이다. GraphD 는 그래프 데이터를 저장하기 위해 분산 준-스트리밍 분할(distributed semi-streaming partition) 기법을 사용한다. 각 머신은 담당할 분할된 정점과 정점의 속성 값, 정점에 인접한 정점들의 리스트(adjacency list)를 할당 받는다.

GraphD 는 질의를 처리하기 위해 송신자-수신자(sender-receiver) 모델을 사용한다. 송신자는 수행된 질의 결과를 이용하여 메시지를 생성하고, 이를 주기적으로 수신자에게 전송한다. 수신자는 송신자로부터 전송된 메시지를 받아 부여된 질의를 수행한다.

GraphD 는 하드웨어 자원을 효율적으로 활용하기 위해 여러가지 기법들을 채택하였다. 우선, 디스크 입출력을 효율적으로 수행하기 위해 연산에 참여하지 않는 간선 데이터를 읽지 않는 방법을 통해 불필요한 디스크 입력을 줄였다. 이를 효율적으로 처리하기 위해 GraphD 는 간선 데이터를 같은 근원 정점을 기준으로 정렬하여 저장하고, 각 정점 별로 이웃한 정점의 수를 메모리 상에 유지한다. 이를 통해 질의에 참여하지 않는 정점을 근원 정점으로 하는 간선 데이터 수만큼 디스크에서 읽지 않는다. 다음으로, 네트워크 트래픽을 줄이기 위해 메시지 조합(message combine)을 구현하였다. 메시지 조합은 네트워크로 메시지를 전송하기전에 가능한 메시지를 하나로 뭉치는 기법이다. 예를 들어, PageRank 에서 인접한 정점으로 랭크 값의 합산을 송신자가 미리 수행한 결과만을 다른 머신으로 전송함으로써 네트워크 트래픽을 줄일 수 있다.

2.2. Chaos

Chaos [4]는 그래프 데이터 전처리 과정을 최소화하고, 확장성에 초점을 둔 디스크 기반의 분산 그래프 처리 시스템이다. Chaos 는 그래프 데이터 저장을 위해 정점과 간선으로 이루어진 그래프 데이터를 임의로 균등하게 각 머신에 분배하는 스트리밍 분할(streaming partition) 기법을 사용한다. 각 머신은 자신이 담당할 분할된 정점과 정점의 속성 값을 할당 받고, 정점 식별자를 근원 정점(source vertex)로 하는 간선 쌍을 할당 받는다. 단순히 정점과 간선의 수를 균등하게 분배하여 저장하기 때문에 Chaos 는 모든 슈퍼스텝마다 모든 간선쌍을 다 읽어야 한다. 이는 막대한 양의 디스크 입출력 발생을 유도 할 수 있다. 그러나 이는 그래프를 분할을 위해 복잡한 전처리 과정이 필요하지 않다는 장점 또한 가지고 있다.

Chaos 는 질의를 처리하기 위해 산포-채집(scatter-gather) 모델을 사용한다. 산포 시기에는 각 머신에 저장된 정점을 모두 메모리에 상주시켜 두고 모든 간선쌍을 스트리밍 방식으로 읽어온 후 다음 슈퍼스텝으로 전달하기 위한 업데이트를 생성한다. 채집 시기에는 산포 시기에 생성된 업데이트를 스트리밍 방식으로 읽어와 질의를 수행한다.

한편, 스트리밍 분할 방식은 머신 간 일의 불균형을 발생시킬 수 있으며, Chaos 는 이를 해결하기 위해 일을 동적으로 분배할 수 있는 작업 가로채기(work-stealing) 기법을 사용한다. 작업 가로채기란 어떤 머신이 할당 받은 일을 끝냈음에도 전체 일은 끝나지 않았을 때, 예상처리 시간과 커뮤니케이션 비용을 고려하여 다른 머신으로부터 작업을 양도받아 대신 처리하여 머신 간의 일의 균형을 맞출 수 있도록 하는 기법이다.

3. 그래프 질의

그래프 데이터의 활용 목적에 따라 다양한 그래프 질의들이 개발되었다. 그래프 질의는 탐색하는 정점의 수에 따라 전체 질의(global query)와 부분 질의(targeted query)로 나뉜다. 전체 질의는 그래프의 모든 정점을 탐색해야 하는 질의를 말하며 페이지랭크, 재시작을 가진 랜덤 워크(random walk with restart)가 대표적이다. 이와 반대로, 부분 질의는 그래프의 일부 정점만 탐색하는 질의를 말하며 너비 우선탐색, k-최근접 이웃(k-nearest neighbor)이 대표적이다. 본 논문에서는 그래프 처리 시스템의 성능평가를 위하여 널리 쓰이는 전체 질의인 페이지랭크와 부분 질의인 너비 우선 탐색을 사용한다.

3.1. PageRank

페이지랭크는 연결이 많은 정점들에게 가중치를 부여하는 그래프 질의로 웹 그래프에서 웹 문서의 중요도를 부여하고자 할 때 많이 쓰인다. 페이지랭크를 그래프 처리 시스템의 관점에서 보면, 한 슈퍼스텝에서 모든 정점이 연산에 참여하여 방대한 양의 메시지를 생성하므로, 네트워크 트래픽 병목 현상이 발생할 수 있다.

3.2. Breadth First Search(BFS)

너비 우선 탐색은 지정된 루트 정점을 시작으로 같은 레벨이 있는 이웃 노드들을 차례대로 탐색하는 그래프 탐색 알고리즘이다. 너비 우선 탐색을 그래프 처리 시스템의 관점에서 보면, 일부의 정점만 한 슈퍼스텝에 참여하므로 필요하지 않은 그래프 데이터를

디스크에서 읽는 과정에서 병목 현상이 발생할 수 있다.

4. 실험

4.1. 실험 환경

실험을 위해 RMAT 합성 그래프 생성 알고리즘 [7]을 사용하여 그래프 데이터 RMAT26 을 생성하였다. RMAT 은 실제 그래프 데이터와 유사하도록 Power-law 법칙을 따르는 그래프를 인위적으로 생성하기 위해 사용하는 방법이다. 정점과 간선의 수는 각각 2^{26} 개, 2^{30} 개이다. 총 32 대의 컴퓨터를 사용하여 실험하였으며, 각 컴퓨터의 환경은 2 개의 Intel Xeon E5-2450 2.1GHz (8 cores per CPU), 32GB DRAM, Intel PCI-E SSD 750, Mellanox QDR Infiniband 네트워크 및 CentOS 7.1 이다. GraphD 와 Chaos 는 모두 C++로 구현되었다. 각 시스템의 성능을 평가하기 위해 3 절에서 설명한 페이지랭크와 너비 우선 탐색 질의를 사용하였다. 페이지랭크는 1 회 반복하였다.

4.2. 실험 결과

4.2.1. 쓰레드 수 증가에 따른 Speed-Up

[sec]		GraphD		Chaos	
# of machines	# of threads	PageRank	BFS	PageRank	BFS
1	1	111.6	142.3	67.8	143.7
	2	77.1	97.2	55.5	145.3
	4	39.2	51.4	56.8	137.9
	8	22.7	26.4	58.4	157.2
	16	22.7	30.1	62.8	136.7

<표 1> 쓰레드 수 증가에 따른 질의 수행 시간

<표 1>은 머신 수를 1 대로 고정시키고, 각 시스템의 쓰레드 수를 증가시켜가며 GraphD 와 Chaos 의 질의 수행 시간을 측정한 표이다. 쓰레드 수를 증가시키면 GraphD 의 질의 처리 속도는 빨라지지만 Chaos 의 질의 처리 속도는 거의 변화가 없다. 그 이유는 Chaos 의 디스크의 입출력을 담당하는 부분이 비효율적으로 설계되었기 때문에, 연산을 담당하는 쓰레드가 늘어나도 디스크 입출력에서 일어나는 병목 현상이 CPU 자원의 충분한 활용을 막았기 때문이다.

4.2.2. 머신 수 증가에 따른 Speed-Up

[sec]		GraphD		Chaos	
# of machines	# of threads	PageRank	BFS	PageRank	BFS
1	16	22.02	30.42	62.8	136.7
2		11.77	17.04	91.8	109.5
4		7.17	9.32	47.2	56.2
8		3.46	5.14	28.2	33.8
16		2.04	3.49	13.1	19.6

<표 2> 머신 수 증가에 따른 질의 수행 시간

<표 2>는 쓰레드 수를 16 개로 고정시키고, 머신 수를 16 대까지 증가시켜가며 GraphD 와 Chaos 의 질의 수행 시간을 측정한 표이다. 머신 수와 질의에 관계없이 GraphD 가 Chaos 보다 항상 빠르게 질의를 처리하고 있다. 이는 Graph 의 연산이 하드웨어 자원의 동시 사용을 통해 효율적으로 수행할 수 있도록 설계되었기 때문이다. 페이지랭크과 너비 우선 탐색에서 각각 GraphD 가 Chaos 보다 평균 6.3 배, 5.8 배 빠른 처리속도를 보였다. 이는 GraphD 가 불필요한 간선 데이터를 읽지 않도록 설계되어 Chaos 보다 효율적으로 디스크 입출력을 수행하기 때문이다. 특히, 페이지랭크에서 GraphD 의 성능이 우수한 것을 볼 수 있다. 이는 GraphD 가 불필요한 데이터를 읽지 않을 뿐 아니라 다수의 머신과 스레드를 충분히 활용할 수 있을 만큼 효율적으로 디스크 입출력을 수행하기 때문이다. 또한, GpradD 는 메시지를 압축하여 보냄으로써 네트워크 병목 문제를 효과적으로 해결하였다.

한편, 두 시스템에서 모두 너비 우선 탐색의 질의 처리 속도가 페이지랭크의 질의 처리 속도보다 느리다. 이는 모든 정점이 참여하여 한번의 슈퍼스텝에 질의가 끝나는 페이지랭크와는 달리 너비 우선 탐색은 일부의 정점이 여러 번의 슈퍼스텝에 걸쳐 참여하여 보다 많은 디스크 입출력이 발생했기 때문이다.

참고문헌

[1] Gonzalez, Joseph E., et al. "PowerGraph: Distributed Graph-Parallel Computation on Natural Graphs." OSDI. Vol. 12. No. 1. 2012.

[2] Malewicz, Grzegorz, et al. "Pregel: a system for large-scale graph processing." Proceedings of the 2010 ACM SIGMOD International Conference on Management of data. ACM, 2010.

[3] Han, Wook-Shin, et al. "TurboGraph: a fast parallel graph engine handling billion-scale graphs in a single PC." Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2013.

[4] Roy, Amitabha, et al. "Chaos: Scale-out graph processing from secondary storage." Proceedings of the 25th Symposium on Operating Systems Principles. ACM, 2015.

[5] Yan, Da, et al. "Effective techniques for message reduction and load balancing in distributed graph computation." Proceedings of the 24th International Conference on World Wide Web. ACM, 2015.

[6] Yan, Da, et al. "Efficient Processing of Very Large Graphs in a Small Cluster." arXiv preprint arXiv:1601.05590 (2016).

[7] Chakrabarti, Deepayan, Yiping Zhan, and Christos Faloutsos. "R-MAT: A recursive model for graph mining." Proceedings of the 2004 SIAM International Conference on Data Mining. Society for Industrial and Applied Mathematics, 2004.