분산 그래프 처리 시스템에 대한 연구 조사

고성윤*, 서인*, 신현규**, 이진수*, 한욱신***
*포항공과대학교 창의 IT 융합공학과
**포항공과대학교 컴퓨터공학과
***포항공과대학교 창의 IT 융합공학과/컴퓨터공학과
e-mail : syko@dblab.postech.ac.kr, iseo@dblab.postech.ac.kr,
yougatup@postech.ac.kr, jslee@dblab.postech.ac.kr, wshan@postech.ac.kr

Survey on Distributed Graph Processing Systems

Seongyun Ko*, In Seo*, Hyungyu Shin**, Jinsoo Lee*, 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

그래프 데이터는 객체와 객체들 간의 관계를 모델링하여 사회 관계망 서비스, 사물 인터넷 그리고 뇌 네트워크등의 데이터를 표현하며 저장한다. 빅데이터의 시대에 빅 그래프를 처리하기 위한 수요는 가파르게 증가하고 있다. 분산 그래프 처리 시스템은 매우 큰 그래프 데이터를 클러스터 내의 여러 머신의 메모리에 나누어 저장함으로써, 빅 그래프의 처리를 가능하게 하였다. 본 논문에서는 최신 분산 그래프 처리 시스템들의 특징들을 비교 연구한다.

1. 서론

그래프 데이터는 객체와 객체들 간의 관계를 표현하며 저장한다. 실제 그래프 데이터는 저밀도성과 정점 간 불균등한 간선 분포를 가진다. Hadoop 이나 Spark 과 같은 일반적인 빅데이터 플랫폼의 처리 방식은, 대규모의 메세징 트래픽이 발생하고 업데이트가 빈번한 그래프 분석 알고리즘들을 처리하는 데에 적합하지 않다. 이에 최근 빅 그래프를 처리하기 위한 다양한 분산 시스템에 대한 연구가 활발히 진행되고 있다.

2. 분산 그래프 처리 시스템

Pregel / Apache Giraph

Giraph [1]는 Pregel [2]의 오픈 소스 버전으로 가장 널리 사용되는 시스템 중의 하나이며 Java 플랫폼 위에 구현되어 있다. 분산 처리 모델로는 Super-step 이라는 지엽적 연산과 그에 따르는 전역적 동기화를 수행하는 Bulk Synchronous Parallel (BSP)을 사용한다. Giraph 는 Hadoop 분산 파일 시스템에 저장된 데이터들을 읽어 메모리에 그래프를 생성한 후 연산을 수행하며, 연산 중간에 발생하는 중간 생성물들은 각 로컬 머신에 저장된다. Giraph 는 정점과 그 정점의 간선들을 같은 머신에 저장하는 정점 분산 기법을 사용한다. 기본적인 정점 할당 정책은 Java 오브젝트를 해쉬한 값을 사용하여 각 정점들이 할당될 머신을 정하는 방법이다. 시스템은 정점들이 어떤 머신에 할당될 지를 결정하는 함수를 유저에게 노출하고 있으므로, 유저가 직접 정책을 결정할 수 있다.

Giraph 는 정점간에 메세지를 주고 받으며 업데이트를 수행하는 Message Passing 분산 처리 모델을 사용한다 (그림 1). 따라서 사용자는 정점 중심의 프로그래밍 인터페이스를 활용해, 직관적인 프로그래밍을 할 수 있다는 장점이 있다 (그림 2). 사용자는 정점들을 State Machine 으로 간주하며 Vertex 클래스를 상속받아 Compute 함수를 구현한다. Compute 함수에서는 각 정점들이 이웃으로부터 받은 메세지를 이용해 자신의 값을 갱신하고, 이어서 자신의 이웃들에게 새로운 메세지를 전달하는 연산을 수행한다.

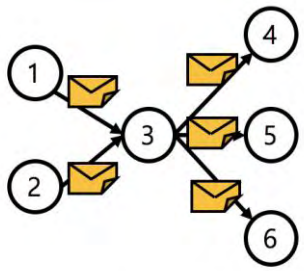

그림 1: 메시지 전달을 이용한 정점 중심 연산 모델.

"이 논문은 삼성전자 미래기술육성센터의 지원을 받아 수행된 연구임" (과제 번호 SRFC-IT1401-04)

```
Template <VertexValue, EdgeValue, MessageValue>
class Vertex {
 public:
  virtual void Compute (MessageIterator* msgs) = 0;
  Iterator<VertexID> GetNeighborIterator();
  void SendMessageTo (VertexID& target_vid,
                      MessageValue& msg);
  void VoteToHalt();
};
```

그림 2: 정점 중심 프로그래밍 인터페이스.

Distributed GraphLab

GraphLab [3]은 CMU 에서 개발된 분산 그래프 처리 시스템으로, C++로 구현되어 있다. GraphLab 은 비동기식의 병렬 처리 방식을 제안하며, Pregel 의 BSP 연산 모델이 가지던 동기화 방식의 비효율성을 개선한다. 비동기식처리에서 발생할 수 있는 정점 데이터의 일관성 문제를 해결하기 위해서 분산 잠금 엔진을 제안한다. 그래프 분산 기법으로는 Giraph 와 동일한 정점 분산 기법을 사용한다.

Giraph 와는 달리 공유 메모리 모델을 사용한다. 공유 메모리 모델에서 유저는 정점간 메세지를 생성하지 않고 이웃 정점의 값을 읽거나 업데이트하는 연산을 수행한다. 사용자 측면에서 마치 메모리 내에 모든 그래프가 있는 것처럼 연산을 수행할 수 있다는 장점이 있다.

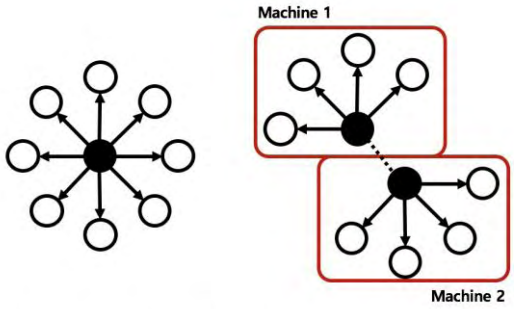

그림 3: 간선 할당 기법; (좌) 원본 그래프 (우) 분산 저장된 그래프.

PowerGraph

PowerGraph [4]는 기존의 GraphLab 시스템을 확장한 시스템이다. 실제 그래프 데이터는 정점들의 간선 수 분포가 Power-law 를 따르기 때문에 극도로 불균등한 특성을 가진다. 따라서 그래프 분산 기법에서 머신간의 정점수와 간선 수를 균등하게 유지하는 것이 중요하다. PowerGraph 는 정점의 간선들을 여러 머신에 나누어 할당함으로써 불균형을 해소하는, 간선 할당 기법을 제안한다 (그림 3). (그림 3)의 머신 1 과 머신 2 는 하나의 정점에 속한 간선들을 나누어 처리하며, 분산 처리된 연산 결과는 모아져서 업데이트 된다.

PowerGraph 는 Gather-Apply-Scatter (GAS) 프로그래밍 모델을 제안한다. GAS 모델은 이웃 노드들에 의한 업데이트를 읽어 들이는 Gather 단계와 이를 이용해 정점의 값을 갱신하는 Apply 단계 그리고 주변 정점의 값을 업데이트하는 Scatter 로 구성된다.

3. 결론

증가하는 그래프 데이터 처리에 대한 수요와 데이터의 규모에 따라서 분산 그래프 처리 시스템들에 대한 연구가 활발히 진행되었다. 본 논문에서는 이러한 시스템들이 구현된 플랫폼, 연산 모델, 프로그래밍 모델, 그리고 그래프 분할 기법에 대해서 분석해보았다.

참고문헌

[1] http://giraph.apache.org/

[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] Low, Yucheng, et al. "Distributed GraphLab: a framework for machine learning and data mining in the cloud." Proceedings of the VLDB Endowment 5.8 (2012): 716-727.

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