Randomized Min-Cut Algorithm

요 약

본 학기말 논문은 Min-cut algorithm의 다양한 랜덤화 버전을 비교하는데 그 목적이 있다. 특히, Kargers randomized Min-cut algorithm과 Karger-Stein randomized Min-cut algorithm을 비교하고자 한다. 아울러, 랜덤화 알고리즘의 성능을 최대화하기 위한 부스팅을 사용하여 그 성능과 시간을 비교한다. 마지막으로, Min-cut이 커뮤니티 탐지에도 사용될 수 있을지 토론한다.

1. 서 론

그래프가 노드 (node)와 엣지 (edge)로 이루어져 있다고 할 때, Min-cut algorithm은 그래프를 두 개의 컴포넌트 (component)로 분할할 때, 제거되는 엣지들의 가중치 합이 최소가 되도록하는 알고리즘이다. 이때, 그래프의 엣지들이 가중치를 가지고 있지 않다면, 최소한의 엣지를 제거하여 두 개의 컴포넌트로 분할하는 알고리즘으로 볼 수 있다. Min-cut algorithm은 최대 유량 문제인 max-flow 문제와 dual 관계에 있음이 알려져 있으며, Min-cut algorithm의 결과로 나오는 cut들의 capacity의 합이 바로 최대 유량이 된다 [1]. 한편, 그래프 G의 노드 집합과 엣지 집합을 각각 V와 E라고 할 때, max-flow 알고리즘은 $O(|V|^2|E|)$ 으로 알려져 있으며, 이를 min-cut 문제를 해결하기 위해 모든 노드 쌍에다가 적용하면, 최대 $O(|V|^4|E|)$의 시간이 소요된다 [2]. 이때, 만약 그래프가 밀집되어 있다면, $O(|V|^6)$의 시간이 소요될 수 있다. 이는 사실 현실적으로 매우 부담이 되는 시간이다. 이에 David Karger는 1993년에 Min-cut을 랜덤화 방식으로 하는 방법을 제안한다 [3]. 이 방법의 기본 아이디어는 각 단계에서 랜덤하게 엣지 하나를 선택하여 그 엣지와 연결된 노드들을 축약 (contraction)해 나가는 것이다. 하지만 이 방법이 정확한 Min-cut을 할 확률은 $\Omega(1/|V|^2)$에 불과하다. 반면, 1996년에 David Karger와 Clifford Stein에 의해 제안된 방법은 전에 David Karger가 제안했던 방법을 다소 수정한 것으로, 정확한 Min-cut을 할 확률이 $\Omega(1/\log(|V|))$으로 많이 향상되었다. 본 논문에서는 앞서 언급된 두 랜덤화 Min-cut algorithm을 소개하고, 성능과 시간을 비교하고자 한다. 그리고 랜덤화 알고리즘의 성능을 최대화하기 위한 부스팅을 사용하여 그 성능과 시간을 비교하고자 한다. 그리고 마지막으로, Min-cut 및 랜덤화 Min-cut의 결과를 통해서 커뮤니티 탐지에도 적용할 수 있을지 토론하며, 그 문제점을 밝히고 개선 방법을 가볍게 소개하고자 한다.

2. 랜덤화 Min-cut 알고리즘

2.1 Kargers randomized Min-cut

Kargers randomized Min-cut 알고리즘은 Min-cut 알고리즘을 근사하기 위한 알고리즘으로써 그 작동 방식은 아래와 같다.

Algorithm Kargers Randomized Min-cut

Input : Graph G(V,E)
Output : the number of minimum cuts

1.Pick an edge from E uniformly at random, say e'∈E.
2.Perform edge contraction**.
3.Repeat steps 1&2 until 2 vertices are left.

**edge contraction : merge two end vertices of e' into a new vertex. and if there are multiple edges between vertices, retain them all.

여기서 시간복잡도를 따져보면, step 1에서 O(1) time, stpe 2에서 $O(|E|)$ time이 소요된다. 사실 step 2는 구현에 따라서 시간복잡도가 많이 달라지는데, 본 논문에서는 $O(|E|)$ time이 소요되도록 알고리즘이 구현되었다. 수업시간에는 $O(|V|)$ time 구현이 가능하다고 배웠지만, 많이 고민을 해보아도 $O(|V|)$ time으로 위 알고리즘을 구현하기는 어려웠다. 따라서 뒤의 실험결과에서도 이 알고리즘의 step 2의 시간 복잡도는 $O(|E|)$ time으로 고려하도록 하겠다. 그리고 step 1과 step 2를 $|V|$-2번 반복해야 하므로 전체적인 시간복잡도는 $O(|V||E|)$가 된다. 한편, 위 알고리즘으로 정확한 min-cut이 도출될 확률은 $\Omega(1/|V|^2)$이다.

2.2 Karger-Stein randomized Min-cut

Karger-Stein randomized Min-cut은 Kargers randomized

Min-cut 알고리즘을 개선한 것으로, 그 작동 방식은 아래와 같다.

Algorithm Karger-Stein Randomized Min-cut

Function : KargerStein(G)
Input : Graph G(V,E)
Output : the number of minimum cuts

if V<4:
    return mincut
else:
    implement the contraction to G to make G1 with 1+|V|/$\sqrt{2}$ remaining vertices.
    implement the contraction to G to make G2 with 1+|V|/$\sqrt{2}$ remaining vertices.
    return min(KargerStein(G1), KargerStein(G2))

Karger-Stein randomized min-cut algorithm의 기본 아이디어는 Kargers Randomized Min-cut algorithm의 iteration이 증가함에 따라 노드의 개수가 줄어들면서 step 1에서 min-cut에 해당하는 엣지를 선택할 확률이 점점 낮아짐을 고려하여, min-cut에 해당하는 엣지를 선택할 확률이 기존에 비해 1/2이 될 때마다 Randomized Min-cut algorithm을 2배로 더 돌리자는 것이다. 실제로 이렇게 하면, 걸리는 시간은 $O(|V|^2\log(|V|))$로 Kargers Randomized Min-cut algorithm에 비해 다소 늘어나지만, 정확한 min-cut이 도출될 확률은 $\Omega(1/\log(|V|))$로 향상된다.

2.3 부스팅

랜덤화 알고리즘의 성능은 확률적이기 때문에, 이를 좀 더 엄밀하게 보장하고자 부스팅 기법을 사용하게 된다. 부스팅 기법이란, 랜덤화 알고리즘을 여러 번 시행하여 그 중에 가장 좋은 출력값을 선택하는 알고리즘으로, 구체적인 방법은 아래와 같다.

Algorithm Boosting

Input : randomized algorithm with success prob p
Output : the best one

Repeat the randomized algorithm $\frac{1}{p}\log(1/\delta)$ times.
Choose the best one.

각 랜덤화 알고리즘의 성공 확률이 p라 할 때, 이를 $\frac{1}{p}\log(1/\delta)$번 반복한 부스팅의 성공 확률은 $\Omega(1-\delta)$가 된다. 한편, Kargers randomized Min-cut과 Karger-Stein randomized Min-cut 의 성공 확률을 똑같이 $\Omega(1/|V|)$로 만들기 위해서 부스팅을 쓴다면, 각각 $O(|V|^2\log(|V|))$, $O(|\log(|V|)^2)$ 번 반복해야 한다. 즉, 성공 확률을 $\Omega(1/|V|)$로 갖기 위해서 부스팅을 사용할 때, Kargers randomized Min-cut의 경우에는 $O(|V|^3|E|\log(|V|))$ 가 소요되고, Karger-Stein randomized Min-cut의 경우에는 $O(|V|^2|\log(|V|)^3)$이 소요되므로, 부스팅 사용 시에는 Karger-Stein randomized Min-cut이 시간적으로 확실한 이득이 있다.

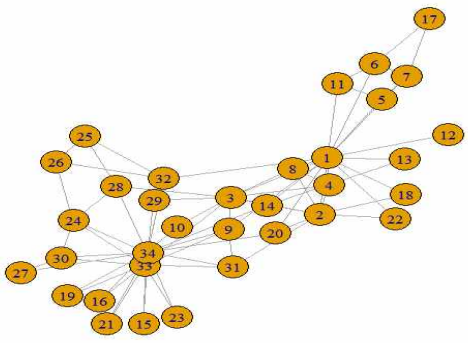
3. 실험

3.1 데이터 셋 및 실험 환경

데이터는 Zachary karate club network [5]를 사용하였다. Zachary karate club network는 34개의 노드와 78개의 엣지로 이루어져 있는 인간 관계 (또는 사람들 간의 친밀도)에 대한 인적 네트워크이며 [5]에서 설명한 배경에 의하면 주어진 인적 네트워크는 2개의 커뮤니티 (파벌)로 구성되어 있다. 자세한 그림은 (그림 1)과 같다. 실험은 8 GB ram, 2.20 GHz, Intel(R) Core(TM) i5-5200U CPU에서 진행되었으며, 프로그램은 R을 사용하였다.

3.2 실험 설계

실험은 Kargers Randomized Min-cut algorithm과 karger-Stein randomized min-cut 알고리즘의 정확도와 소요시간을 측정하기 위해서 100번씩 각 알고리즘을 실행하였다. 그리고 그 후에 두 랜덤화 Min-cut 알고리즘에 부스팅을 적용하여 100번 반복하여 정확도와 소요시간을 측정하였다.



(그림 1) Zachary Karate Club Network 그래프 개형

3.3 실험 결과

(그림 1)을 보면, 진짜 Min-cut은 1임을 쉽게 알 수 있다. (예를 들면 1번과 12번 사이의 선을 끊으면 12번은 다른 노드들과

분리된다.) 한편 Kargers Randomized Min-cut algorithm과 arger-Stein randomized Min-cut algorithm을 비교한 실험 결과는 (그림 2), (그림 3) 과 같다. (그림 2)와 (그림 3)을 보면 Kargers Randomized Min-cut algorithm은 Karger-Stein randomized Min-cut algorithm에 비해 속도가 빠르지만, 정확도가 다소 떨어지는 것을 확인할 수 있다. 실제로, Kargers Randomized Min-cut algorithm은 평균적으로 0.00148947초 걸리고 정확도 (100번 중에 Min-cut이 1이라고 판단한 비율)는 26%에 불과하지만, Karger-Stein Randomized Min-cut 알고리즘은 평균적으로 0.03131242초 걸리고, 정확도 (100번 중에 Min-cut이 1이라고 판단한 비율)는 98%에 육박한다. 한편, 성공 확률을 $\Omega(1/|V|)$로 갖도록 부스팅을 적용해본 실험 결과는 (그림 4), (그림 5)와 같다. 이때는 두 랜덤화 방법 모두 정확도 100%를 가지지만, 걸리는 시간 면에서 Karger-Stein Randomized Min-cut이 확실히 빠르다. 실제, Kargers Randomized Min-cut algorithm의 경우 평균적으로 4.11356초, Karger-Stein Randomized Min-cut algorithm은 0.3798922초 소요되었다. 한편, 커뮤니티 탐지에서 Min-cut이 효과적이지 않음을 (그림 1)을 통해서 알 수 있다. 커뮤니티의 특징을 좀 더 가지도록 Min-cut을 하기 위해서는 [6]과 같은 방법을 고려하는 것을 생각해볼 수 있을 것이다.

결론

Min-cut 알고리즘을 근사한 다양한 랜덤화 방법들을 비교실험 하였다. Karger-Stein algorithm이 Kargers (그림 2) 두 랜덤화 Min-cut 알고리즘의 시간 비교 알고리즘에 비해 느리지만, 성능이 좋으며, 부스팅 시에는 같은 성능 대비 걸리는 시간이 훨씬 적음을 확인하였다. 그리고 Min-cut 알고리즘이 커뮤니티 탐지에는 효과적이지 않음을 확인하였다.

참고문헌

[1]https://en.wikipedia.org/wiki/Max-flow_min-cut_theorem

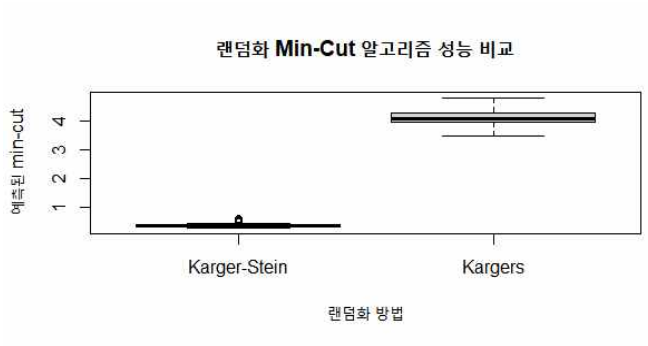
[2]http://courses.csail.mit.edu/18.337/2016/final_projects/matthew_lindsay/Final+Presentation.html

[3]David R. Karger, “Global Min-cuts in RNC, and Other Ramifications of a Simple Min-Cut Algorithm”, Proceedings of the Fourth Annual {ACM/SIGACT-SIAM} Symposium on Discrete Algorithms, 25-27 January 1993, Austin, Texas, {USA}, pp. 21-30, 1993

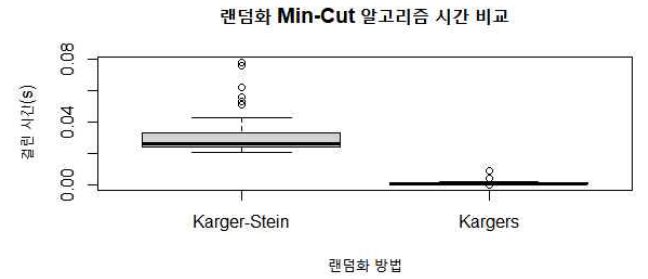
[4]David R. Karger and Clifford Stein, “A New Approach to the Minimum Cut Problem”, J. {ACM}. Vol. 43, No. 4, pp. 601-640, 1996

[5]W. W. Zachary, “An information flow model for conflict and fission in small groups”, Journal of anthropological research, pp.452-473, 1977.

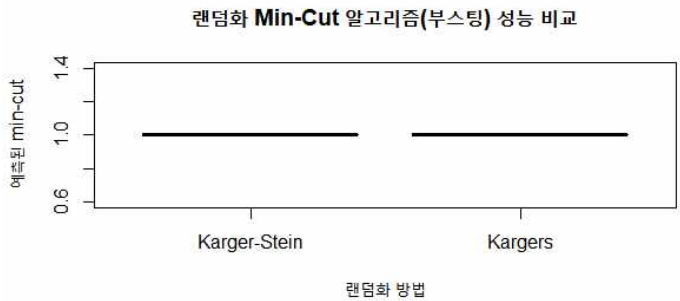
[6]Guangmo Tong, Lei Cui, Weili Wu, Cong Liu, and Ding{-}Zhu Du, “Terminal-set-enhanced community detection in social networks”, 35th Annual {IEEE} International Conference on Computer Communications, {INFOCOM} 2016, San Francisco, CA, USA, April 10-14, 2016



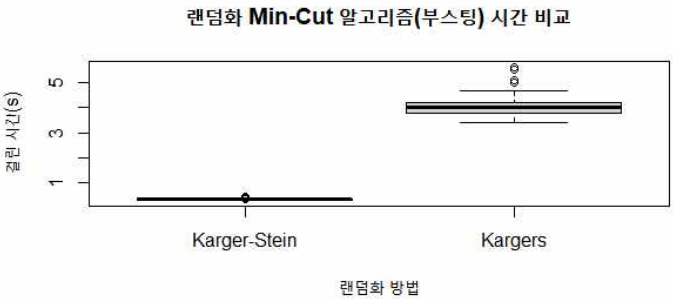
(그림 2) 두 랜덤화 Min-cut 알고리즘의 성능 비교



(그림 3) 두 랜덤화 Min-cut 알고리즘의 시간 비교



(그림 4) 부스팅을 적용한 알고리즘의 성능 비교



(그림 5) 부스팅을 적용한 알고리즘의 시간 비교