일반 그리드 그래프를 위한 입출력 효율적인 영역 구분자

허준호° R.S. Ramakrishna
광주과학기술원 정보통신공학과
{jhher°, rsr}@gist.ac.kr

I/O Efficient Cycle Region Recognizer for General Grid Graphs

Jun-Ho Her° R.S. Ramakrishna
Dept. of Inf. & Comm., Gwangju Institute of Science and Technology (GIST)

요 약

자료의 크기가 방대한 응용 프로그램에서는 메인 메모리와 저장 장치간의 자료 입출력(I/O)이 전체 계산의 주요 병목 요인으로 작용한다. 본 논문은 [2004 한국정보처리학회 추계논문집 제11권 제2호 1139-1142]에서 제안된 그리드(grid) 그래프를 위한 입출력 효율적인 depth-first search (DFS) 알고리즘을 지원하기 위한 입출력 효율적인 영역 구분자를 구하는 알고리즘을 제안 한다. 그 입출력 복잡도(I/O-complexity)는 O(sort(N))이다. 여기서 $N=|V|+|E|$이고 $sort(N)=\theta((N/B)log_{M/B}(N/B))$이다.

1. 서 론

저장장치(하드 디스크)의 성능은 계산장치(CPU 및 메모리)의 속도에 비해 매우 느릴 뿐 더러 점점 그 격차가 커지고 있는 실정이다[1]. 이에 최근에 알고리즘 분야에서도 계산 스텝뿐만 아니라 입출력 횟수를 고려하기 위한 움직임이 활발히 일고 있다[2-9]. 본 논문에서는 저장장치를 고려하는 두 단계 입출력 모델 (two-level I/O model)[10]을 바탕으로 알고리즘을 설계하고 분석한다. 이 모델에서의 파라미터는 다음과 같다:

N= 문제의 크기 (그래프 문제에서는, $N=|V|+|E|$),
M= 내부 메모리의 크기,
B= 디스크 블록의 크기,
D= 독립적인 디스크 드라이브의 수,

여기서 MKN 이고 $1 \le DB \le M/2$ 이다. 본 논문에서는 편의상 D를 1로 둔다. 두 단계 입출력 모델에 대한 보다 자세한 사항은 [10]을 참고하기 바란다.

저자들은 [11]에서 그리드(grid) 그래프를 위한 입출력 효율적인 DFS 알고리즘을 제안 하였다. 여기서, 그리드 그래프는 무한 이차원 정수 격자상의 유한한 점유도(node-induced) 그래프로 정의 된다. 일반화된 그리드 그래프는 대각 edge가 존재하고 서로 교차 할 수 있으므로 비평면 그래프로 분류 된다. 제안 알고리즘은 평면 그래프를 위한 알고리즘이 여전히 동작할 수 있게끔 부가적인 동작을 첨가 한 것인데, 여기서 핵심이 되는 것은 내부 점과 외부 점을 특정 비율로 나누어 주는 simple cycle 분할자를 찾는 것이다. 본 논문은 이를 위한 전초 단계로써 simple cycle 영역 구분자를 찾아주는 알고리즘을 제안한다.

본 논문은 다음과 같이 구성된다. 2장에서는 관련 알고리즘에 대해 살펴보고, 3장에서는 제안 알고리즘에 대해 기술하며, 마지막으로 4장에서는 결론을 맺는다.

2. 관련 알고리즘

[11]의 알고리즘은 평면 그래프를 위한 입출력 효율적인 DFS 알고리즘과 유사하게 진행되면서 그리드 그래프의 비 평면성으로 인해 발생하는 cycle separator의 비분할 요인을 내부 루프에서 제거함으로써 비 평면 그리드 그래프에 대해서도 'divide and conquer' 방법을 가용케 한다. 이러한 알고리즘을 지원하기 위해서는 그리드 그래프를 위한 simple cycle 영역 구분자를 구해야 하는데 이는 평면 그래프에 대한 simple cycle 2/3-separator 알고리즘을 이용하여 구할 수 있다.

평면 그래프에 대한 simple cycle 2/3-separator C는 그 내부나 외부에 2/3|V|이상의 vertex들을 포함하지 않도록 하는 cycle이다. 다음은 평면 그래프에 대한 simple cycle 2/3-separator C를 구하는 알고리즘에 대해 간략히 기술한다.

1. 큰 face들을 체크한다: 경계가 최소한 1/3|V|의 크기를 가지는 face가 있는지 체크 한다 (그림 1(a)). 주어진 그래프 G의 각 face를 경계에 따라 있는 vertex들의 리스트로 나타낸다. 이러한 표현을 하는데 $O(sort(N))$ 입출력이 소요된다[12]. 만일 그런 face가 발견되면, 그 것의 경계를 C로 보고하고 끝낸다.
2. 큰 subtree들을 체크한다: 만일 1단계가 통과되면, 주어진 그래프 G의 듀얼 그래프 G^*에 대한 spanning tree T^*를 구하고 임의의 vertex r을 근으로 정한다. T^*의 모든 vertex v에 대해서 $T^*(v)$는 v를 중심으로 한 subtree에 대한 표현이고 이 subtree의 vertex들은 face들에 대응되므로 $G(v)$는 그 face들의 경계들로 구성된 그래프로 정의한다. [13]에서는 $G(v)$의 경계가 simple cycle임을 증명하였고 $1/3|V| \le |G(v)| \le 2/3|V|$인 v를 찾는 방법을 보이고 있다 (그림 1(b)). 이것이 성공하면 $G(v)$의 경계를 C로 보고하고 끝낸다.
3. 한 큰 subtree를 나눈다: 만일 1,2단계가 실패하였다면, $|G(v)| > 2/3|V|$ 이고 v의 모든 자식들

$w_1, \ldots, w_k$ 에 대해서 $|G(w_i)| < 1/3|V|$인 vertex v가 존재한다고 할 수 있다. [13]에서 저자들은 $G(v)$의 face들 v^*의 경계와 그래프 $G(w_1), \ldots, G(w_k)$의 한 부분집합으로 구성되는 $1/3|V| \le |G'| \le 2/3|V|$인 한 subgraph G'를 구하는 것과, G'의 경계가 한 simple cycle임을 보이고 있다.(그림 1(c)).

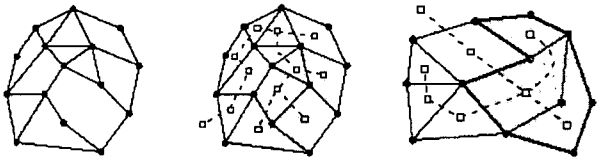

그림 1. 평면 그래프에서 simple cycle 2/3-구분자

다음 정리는 이 알고리즘에 대한 것이다.

정리 1[13] 어떤 *biconnected* 평면 그래프의 *한 simple cycle 2/3-separator 는 O(sort(N)) 입출력 복잡도와 선형공간으로 구할 수 있다.*

3. 한 simple cycle 2/3-영역 구분자 C 구하기

제안 알고리즘의 핵심 아이디어는 주어진 그리드 그래프의 평면 subgraph를 가장자리 cycle을 통해서 구하고 이 subgraph에 대해서 2절의 평면 그래프에 대한 알고리즘(planar$_{2/3}$SCS)을 적용하여 simple cycle C를 구하는 것이다. 그 절차는 다음과 같다:

1. 가장자리 cycle C_{out}를 구한다: 가장자리라 함은 기하적인 위치에 의해 가장 바깥쪽임을 뜻한다.
   가. G에서 가장자리에 있는 vertex들로 구성된 집합을 V_{out}라하자 (그림 2(b)에서 회색 vertex들). 모든 $v \in V_{out}$에 대해서, 만일 v가 한 edge e와 incident하다면, e는 E_{out}(그림 2(b)에서 두꺼운 선들)에 포함된다. 그리고 만일 e의 다른쪽 끝 vertex가 V_{out}에 포함되지 않았다면 V_{out}에 포함시킨다(그림 2(c)에서 검은 vertex들).
   나. $G_{out}=(V_{out}, E_{out})$의 모든 edge $\{v,w\}$를 양 방향성 edge들 (v,w) 와 (w,v)로 교체한다(그림 2(c)).
   다. 결과 양방향 subgraph D_{out}으로부터 한 가장자리 cycle C_{out} 추출 하기(그림 3): D_{out}의 시계방향으로 순회하면서 임의의 한 cycle을 찾고, 반복적으로 더욱 바깥쪽에 있는 subpath의 유무를 체크하고 존재하면 C_{out}의 해당 subpath를 이 subpath로 갱신 한다.
   라. cycle C_{out}의 정제: 만일 구한 cycle C_{out}이 교차 edge를 갖고 있으면, 교차 edge의 각 끝 vertex 위치를 교환하고 이로 인한 비-cycle edge들의 새로운 교차를 제거 하기 위해 그 비-cycle edge를 제거 한다(그림 4(a)).
2. 평면 subgraph G' 추출하기(그림 4(b))
   가. cycle C_{out}을 가로지르는 edge들을 제거한다.
   나. cycle C_{out}내의 모든 교차 대각 edge쌍에서 한 edge씩 제거 한다.
3. planar$_{2/3}$SCS을 적용한다(그림 4(c)): G'에 대해서 planar$_{2/3}$SCS을 적용한다. 여기서 단계 1.라와 단계 2의 edge제거에 의해서 한 face내에 고립 vertex (또는 connected component)가 존재 할 수 있는데(그림 2), planar$_{2/3}$SCS를 적용할 때 한 face f의 고립 vertex는 f의 vertex cardinality에 기여를 하게끔 하고 C_{out}의 바깥쪽 vertex들도 전체 vertex cardinality $|V|$에 고려되게 한다.

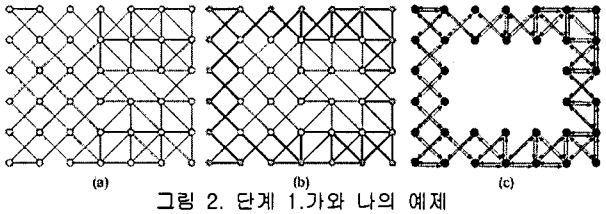

그림 2. 단계 1.가와 나의 예제

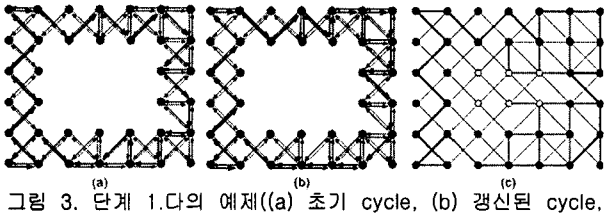

그림 3. 단계 1.다의 예제((a) 초기 cycle, (b) 갱신된 cycle, (c) 최종 cycle)

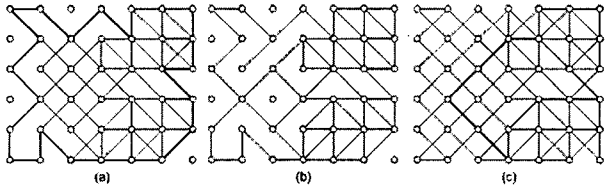

그림 4. 단계 1.라, 단계 2 그리고 단계3의 예제

보조정리 2 *Biconnected 그리드 그래프의 한 simple cycle 2/3-영역 구분자는 O(sort(N)) 입출력을 통해 구할 수 있다.*

증명. *타당성*: 단계 1은 [12]의cycle인식과 유사하면서 가장자리 vertex들과 그 incident edge들 그리고 그 edge들의 다른 한 끝 vertex들로 구성된 subgraph에 적용한다. 또한 가장자리 vertex들의 biconnectivity는 여전히 보존되므로 최소 하나의 cycle이 이 subgraph에 존재하게 된다. 따라서 우리는 G의 가장자리 cycle을 얻을 수 있다. 단계 2에서 우리는 모든 edge교차를 없앴기 때문에, 단계 3의 planar$_{2/3}$SCS를 적용할 수 있다. 또한 우리는 한 face내의 고립 vertex가 여전히 vertex cardinality에 기여 할 수 있게끔 하였고 C_{out}의 바깥쪽 vertex들도 전체 $|V|$에 고려되게 하였으므로, 2/3$|V|$값

을 제대로 반영하게 된다.

복잡도: 그리드 그래프에서 vertex는 흔히 2차원 좌표에 의 해 인식되므로 단계 1.가의 가장자리 vertex들은 x와 y좌표에 의한 vertex정열로부터 구할 수 있고 단계 1.가의 나머지는 가장자리 vertex들과 그 incident edge들을 scan하면되므로 단계 1.가는 $O(sort(N))$의 입출력이 소요된다. 단계 1.나와 단계 1.라는 간단히 scan에 의해 처리 가능하므로 $O(scan(N))$의 입출력이 소요된다. 단계 1.다에서, 처음 cycle에서 몇 몇 scan을 통해서 C_{out}을 찾을 수 있으므로 단계 1은 전체적으로 $O(sort(N))$의 입출력이 소요된다. 단계 2는 C_{out}와 그 내부를 간단히 scan하면서 수행 할 수 있다. 단계 3은 정리 1에 의해 $O(sort(N))$의 입출력이 소요되므로 전체 입출력 복잡도는 $O(sort(N))$이 된다. □

4. 결 론

비 평면 그래프인 일반 그리드 그래프에서 안과 밖을 2/3으로 나누는 cycle을 입출력 효율적으로 구하는 방법을 제안 하였다. 이 방법은 depth-first search와 같은 다른 복잡한 알고리즘에 병렬성 및 입출력 효율성을 제공해주는 역할을 할 수 있다. 그리드 그래프와 같이 비 평면성이지만 어느 정도 거시적으로 평면성을 내포하고 있는 대부분의 그래프 종류에서도 수행 가능할 것으로 보인다. 제안 알고리즘이 디스크 그래프와 같은 다른 비 평면 그래프 종류에 대해서도 타당성이 있는지 검토하는 것이 향후 과제이다.

참고문헌

[1] Jeanna Neefe Matthews, "Improving File System Performance With Adaptive Methods," Ph.D. Dissertation, University of California Berkely, December 1999.

[2] Y. J. Chiang, M. T. Goodrich, E. F. Grove, R. Tamassia, D. E. Vengroff, and J. S. Vitter, "External-memory graph algorithms," In Proc. of the 6th Annual ACM-SIAM Symposium on Discrete Algorithms, pp 139-149, 1995.

[3] V. Kumar and E. J. Schwabe, "Improved algorithms and data structures for solving graph problems in external memory,"In Proceedings of the 8th IEEE Symposium on Parallel and Distributed Computing, pp. 169-177, 1996.

[4] K. Munagala and A. Ranade, "I/O-Complexity of graph algorithms," In Proc. of the 10th Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 687-694, 1999.

[5] A. Buchsbaum, M. Goldwasser, S. Venkatasubramanian, and J. Westbrook, "On external memory graph traversal,"In Proc. of the ACM-SIAM Symposium on Discrete Algorithm, pp566-575, 2000.

[6] L. Arge, L. Toma, and J. S. Vitter, "I/O-Efficient Algorithms for Problems on Grid-based Terrains,"Journal of Experimental Algorithms, vol. 6, page 1, 2001.

[7] U. Meyer, "External Memory BFS on Undirected Graphs with Bounded Degree," In Proc. 12th ann. Symposium on Discrete Algorithms, pp. 87-88, ACM-SIAM, 2001.

[8] A. Maheshwari and N. Zeh, "I/O-optimal algorithms for planar graphs using separators," In Proc. of the 13th annual ACM-SIAM Symposium on Discrete Algorithms, pp. 372-381, 2002.

[9] A. Maheshwari and N. Zeh. "I/O-Efficient Algorithms for Outerplanar Graphs." Journal of Graph Algorithms and Applications, 8(1):47-87,, 2004.

[10] A. Aggarwal and J. S. Vitter, "The input/output complexity of sorting and related problems," Communications of the ACM, pp. 1116-1127, September 1988.

[11] 허준호 R.S. Ramakrishna, "그리드 그래프를 위한 입출력 효율적인 Depth-First Search 알고리즘," 2004 한국정보처리학회 추계학술대회논문집, 제11권, 제2호, pp.1139-1142, 2004.

[12]David Hutchinson, Anil Maheshwari, and Norbert Zeh, "An external memory data structure for shortest path queries," Discrete Applied Mathematics, vol.126, no.1, pp.55-82, March 2003

[13] L. Arge, U. Meyer, L. Toma, N. Zeh, "On External-Memory Planar Depth First Search," *Journal on Graph Algorithms and Applications*, vol. 7, no. 2, pp. 105-129, 2003.