Solid Grid 그래프를 위한 입출력 효율적인 Depth-First Search 알고리즘

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

External-Memory Depth-First Search Algorithm for Solid Grid Graphs

Jun-Ho Her°, R. S. Ramakrishna
Dept. of Inf. & Comm., Kwang-ju Institute of Science and Technology (K-JIST)

요 약

여러 과학 및 공학 응용 프로그램에서 다루는 그래프 데이터는 종종 그 크기가 너무 커서 컴퓨터의 주 메모리에 다 들어 갈 수 없는 경우가 많다. 이러한 방대한 크기의 자료를 처리하면서 입출력의 빈도가 자연적으로 커지게 되고 전체 계산에서 주요한 병목 요인으로 작용한다. 본 논문은 solid grid 그래프를 위한 입출력 복잡도 (I/O-complexity)가 O(sort(N))인 depth-first search (DFS) 알고리즘을 제안한다. 여기서, N=|V|+|E| 이고 sort(N)=$\Theta((N/B)\log_{M/B}(N/B))$ 이다. 이 전까지 알려진 가장 좋은 알고리즘은 적절한 sub-grid 입출력을 바탕으로 한 전통적 DFS 알고리즘으로 그 입출력 복잡도는 $O((N/B)B^{1/2})$ 이다.

1. 서 론

최근 컴퓨터 산업에서 저장장치의 성능은 계산의 속도를 못 따라가는 실정이다. 더구나, 그 격차는 오히려 커지고 있는 추세이다. 지난 90대 초부터 이러한 현상에 주목하고 알고리즘 차원에서 이러한 문제에 대응하기 위한 움직임이 일기 시작했다. 입출력 효율적 (I/O-efficient) 알고리즘 또는 외부 메모리 (external-memory) 알고리즘이라고 불리는 알고리즘 개발의 요지는 입출력 빈도를 줄여 (주 메모리에 다 넣을 수 없는) 큰 데이터를 다루는 계산에서 전체 수행 성능 저하의 근본 원인을 줄이는 데 있다. 특히, 그래프를 다루는 알고리즘에 대한 연구가 활발히 진행 되어 왔지만[1]-[9], 아직 많은 문제들이 풀리지 않고 있다.

일반적인 그래프에 대한 알려진 입출력 효율적인 DFS 알고리즘의 복잡도는 $O(|V|+|V|\cdot M^{-1}\cdot scan(|E|))$ I/Os [1] 또는 $O((|V| + scan(|E|))\cdot \log_2|V|)$ [2]이고 Kumar의 알고리즘은 edge의 수가 vertex수에 비해서 어느 정도 큰 경우(즉, $|E| \geq |V|$)에는 더욱 효율적이다. 특정 그래프 클래스에 대해서는 그래프의 구조적 특징을 활용하는 좀 더 효율적인 알고리즘들이 개발되어 왔다. 평면 (embedded planar) 그래프에 대해서는 O(sort(N))의 입출력 복잡도를 가지는 알고리즘이 개발 되었다 [8]. 일반적인 grid 그래프 (grid 그래프는 몇몇 대각선이 교차되면서 nonplanar일 수 있다)에 대해서는 기존의 전통적 DFS 알고리즘에 바탕 하여 적절한 subgrid 입출력이 이루어지도록 수정한 알고리즘이 있으며[6], 그 복잡도는 $O((N/B)B^{1/2})$이다.

본 논문은 solid grid 그래프라고 하는 특수한 grid 그래프 클래스에 대한 입출력 효율적인 DFS 알고리즘을 제안한다. Grid 그래프는 무한 이차원 정수 격자상의 유한한 점유도(node-induced) 그래프로 정의 된다. Solid grid 그래프는 그래프 내부가 꽉 차 있어 내부의 격자에는 항상 edge가 존재하는 grid 그래프의 한 종류이다 (그림 1 참조). Solid grid 그래프의 구조적인 특징을 활용하여 이러한 그래프 클래스에 한 하여

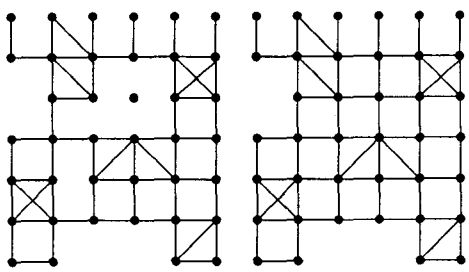


그림 1 일반 grid 그래프 (a) 와 solid grid 그래프 (b)

O(sort(N))의 입출력 복잡도를 가지는 알고리즘을 제안한다.

본 논문은 다음과 같이 구성된다. 2장에서는 예비지식과 관련 알고리즘에 대해서 기술하고, 3장에서는 제안 알고리즘에 대해서 기술하며, 마지막으로 4장에서는 결론을 맺는다.

2. 예비 개념과 평면 그래프 DFS 알고리즘

본 논문에서는 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 M<N 이고 $1 \leq DB \leq M/2$ 이다. 본 논문에서는 편의상 D

를 1로 둔다. 두 단계 입출력 모델에 대한 보다 자세한 사항은 [10]을 참고하기 바란다.

Grid 그래프에서 각 vertex는 보통 이차원 좌표 (i,j)로 표현되며 (i, j), (i+1, j), (i, j+1), 그리고 (i+1, j+1)로 이루어진 유도 그래프(induced subgraph)를 grid의 한 cell이라고 부른다. 또한, DFS-tree는 cross edge를 가지지 않는 spanning tree이며 normal spanning tree라고도 불린다.

1장에서 언급한 평면 그래프 DFS 알고리즘(Planar_DFS)은 제안 알고리즘 상에서 중간 단계로써 활용된다. Planar_DFS의 주요 절차는 다음과 같다. 첫째, 평면 그래프의 각 face들을 루트가 포함된 face를 기점으로 여러 단계의 층으로 나누어 고려한다. 둘째로, 각 단계의 층으로부터 바깥 평면 그래프(outerplanar graph)들을 얻는다. 각 바깥 평면 그래프의 DFS-tree들을 구한 후 최종적으로 이들 DFS-tree를 합하면 주어진 평면 그래프의 DFS-tree를 얻는다. 보다 자세한 사항은 [8]을 참고하기 바란다.

보조정리 1 [8] *평면 그래프의 DFS-tree는 O(sort(N)) 입출력으로 구할 수 있다.*

3. 제안 알고리즘

제안 알고리즘의 주요 절차는 다음과 같다. 주어진 solid grid 그래프를 입출력 효율적으로 평면 그래프로 변형한 후 기존의 Planar_DFS 알고리즘을 이용하여 변형된 그래프의 DFS-tree를 구한다. 그 후 원래의 그래프로 되돌리면서 DFS-tree를 완성하는 것이다. 다음의 각 절에서 자세한 사항을 기술한다.

3.1 평면 그래프로의 변형

루트가 r로 주어진 한 solid grid 그래프 G에 대해서:

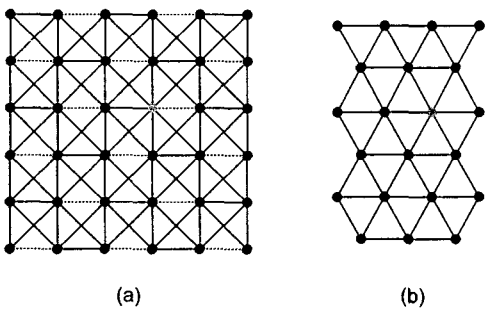
1. G의 각 cell에 대해서,
   (ㄱ) 교차 대각 edge 유무를 파악한다.
   (ㄴ) 교차 대각 edge를 가지는 cell들에 대해서 cell 내의 한 수평(또는 수직) edge를 마크(mark)한다.

이 단계는 입력 그래프의 스캔을 통해서 수행 가능하므로 O(scan(N))의 입출력 복잡도를 가진다.

2. 마크된 edge들을 압축(contraction)한다.

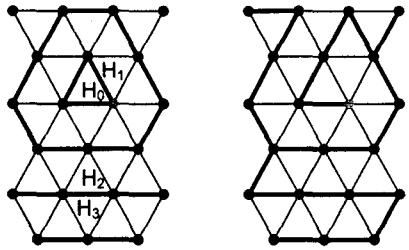
이 단계 역시 O(scan(N))의 입출력 복잡도를 가진다.

각 단계별로 O(scan(N))의 복잡도를 가지므로 평면 그래프 변형 알고리즘 전체적으로 O(scan(N))의 입출력 복잡도를 가진다. 이 것은 보조정리 2의 입출력 복잡도 부분을 증명한다.



(a) (b)

그림 2 평면화 알고리즘 적용 예



(a) (b)

그림 3 Planar_DFS 적용의 예

보조 정리 2. *위의 알고리즘은 한 solid grid 그래프로부터 O(scan(N))의 입출력을 통해 평면 그래프(embedded planar graph)를 얻는다.*

증명. 위의 알고리즘은 한 solid grid 그래프에 대한 모든 K4형태의 cell을 K3형태로 변형하므로 그 결과 그래프는 이미 평면화된 그래프이다. □

그림 2는 한 solid grid 그래프에 대한 예를 보인다.

3.2 Planar_DFS 단계

이 단계에서는 기존의 평면 그래프를 위한 DFS 알고리즘을 앞서 변형된 그래프에 대해 그대로 적용한다. 한 solid grid 그래프 G(N=|E|+|V|)가 G'(N'=|E'|+|V'|)으로 변형되었다고 할 때, N과 N'의 관계는 N'=αN (0<α<1)이고 이와 함께 보조 정리 1로부터 다음의 따름 정리 1을 얻을 수 있다.

따름 정리 1. *크기가 N으로 주어진 solid grid 그래프 G대해 평면화 알고리즘에 의해 평면화된 그래프의 DFS-tree는 O(sort(N))의 입출력으로 구할 수 있다.*

그림 3에서 그림 2의 예제에 대하여 Planar_DFS를 적용한 경우를 보인다.

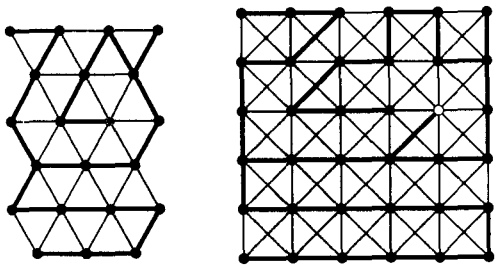
3.3 원래 그래프의 DFS-tree 구하기

이 알고리즘에 대해서는 논문의 지면 제한 관계로 함축적으로 기술한다. 만일 평면 그래프 변형에서 압축된 한 edge가 루트 r을 한 끝 점으로 할 때, 다른 끝 점을 r"이라 하자. G에서 r"이 제거된 (incident edge들도 제거된) 그래프를 G"이라고 한다면 G'의 DFS-tree T'로부터 G"(=G-r")의 DFS-tree T"을 적절한 simple path (TSSP) 복구 과정을 되풀이 함으로써 구할 수 있다. 이러한 simple path (TSSP)에 대한 정의와 복구 과정에 대해서는 [11]을 참조하기 바란다. 마지막으로 r"에 대한 한 DFS-tree edge를 찾아 줌으로써 G의 DFS-tree T를 최종적으로 얻는다.

DFS-tree 구성 알고리즘은 다음과 같다:

1. G'를 원래의 그래프 G로 회복;

전통적인 edge-contraction 및 복구 방법을 통해서 G을 회복한다. 회복하는 동안 각 DFS-tree정보를 G상에 표기한다. 이 과정은 O(scan(N))의 입출력을 소모한다. 왜냐하면, 그래프 회복은 스캔 입출력을 통해서 가능하므로 O(scan(N))의 입출력이 소모되고 추가적으로 T'의 정보를 표기하는데 O(scan(N)) 의 입출력이 소모되기 때문이다.



(a) (b)

그림 4 최종 DFS-tree를 얻는 예

2. T'의 TSSP에 대응되는 T"의 모든 TSSP들을 완성;
[11]에 설명된 TSSP 복구 알고리즘을 모든 T'에 대해서 적용함으로써 T"을 얻는다. 이 과정은 각 TSSP들 대한 입출력 복잡도의 합이므로 $\Sigma_{ij}O(scan(|P"_{ij}|))=O(scan(N))$의 입출력이 소모 된다.
3. r"에 대한 DFS-tree edge를 구함;
입출력 입장에서 이것은 r" 부근의 데이터에 대한 입출력만으로 충분하므로 O(1)의 입출력이 소모된다. 전체적으로 O(scan(N))의 입출력이 소모된다.
그림 4는 앞선 예제 그림에 대해서 최종 DFS-tree를 얻는 예를 보인다.

보조 정리 3. *제안 알고리즘으로부터 얻은 그래프 T는 G의 DFS-tree이다.*

증명. T"이 DFS-tree이라면 T는 자연스럽게 DFS-tree이므로 T"가 G"의 DFS-tree임을 보인다.

$V_{G'cont}$가 G'상에서 G에 대한 각 edge contraction의 결과로 생긴 vertex 집합이라고 할 때, $|V_{T'}|=|V_{G'}|=|V_G|-|V_{G'cont}|$ 이고 $|E_{T'}|=|V_{T'}|-1$ 이다. 왜냐하면 T'은 증명된 Planar_DFS 알고리즘에 의해 이미 DFS-tree이기 때문이다. 따라서, T'에서 T"로의 vertex 증가 수는 $|V_{G'cont}|-1$ 이므로 $|V_{T'}| = |V_T|+|V_{G'cont}|-1 = |V_{G'}|$ 이다. T'에서 T"로의 edge 증가 수도 역시 $|V_{G'cont}|-1$ 이므로, $|E_{T'}| = |E_T| + |V_{G'cont}|-1 = |V_{T'}| + |G_{V'cont}|-2 = |V_{T'}|-1$ 이다. T"는 T'의 확장으로부터 얻어진 그래프이므로 T"는 연결되어있다(connected). 따라서, T"는 G"의 한 spanning tree 이다.

주어진 normal spanning tree T'에 대해서, 3.3절의 알고리즘을 적용한 G"의 spanning tree T" 역시 normal하다. T"에 대해서 G"상에 한 cross-edge e"가 존재한다고 가정하면 e"에 대응되는 G'의 edge e' 또한 cross-edge 이다(이에 대한 상세한 증명은 [11]을 참조바람). 이것은 T'의 normality에 대해서 모순이다. 따라서, T"에는 cross-edge가 존재하지 않는다. □

각 절의 입출력 복잡도와 보조 정리 3으로부터 우리는 본 논문의 주요 결과인 정리 1을 얻을 수 있다.

정리 1. *임의의 solid grid 그래프의 DFS-tree는 O(sort(N))의 입출력으로 구할 수 있다.*

4. 결 론

본 논문은 solid grid 그래프를 위한 새로운 입출력 효율적인 DFS 알고리즘을 제안하였다. 제안 알고리즘은 solid grid 그래프 클래스에 대한 DFS-tree를 얻는데 있어서 입출력 복잡도 측면에서 지금까진 알려진 알고리즘 중 가장 좋은 성능을 보인다. 하지만 그와 유사한 입출력 성능이 일반적인 grid 그래프에서도 보장 될지는 아직 미결 과제이다. 또한 bounded treewidth 그래프와 같은 다른 특정 그래프 클래스에 대해서도 입출력 효율적인 알고리즘을 개발하는 것이 당면 과제이다.

5. 참고 문헌

[1] 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.

[2] 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.

[3] 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.

[4] 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.

[5] 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.

[6] 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.

[7] 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.

[8] 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.

[9] L. Arge, L. Toma, N. Zeh. "I/O-Efficient Topological Sorting of Planar DAGs." In *Proceedings of the 15th ACM Symposium on Parallelism in Algorithms and Architectures*, pp. 85□ 93, 2003.

[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] Jun-Ho Her and R. S. Ramakrishna, " External-Memory Depth-First Search Algorithm for Solid Grid Graphs," Technical report, K-JIST, November 2003.