그리드 구조에서 개선된 가쉬핑 알고리즘

이 현 섭*, 좌 경 룡

한국과학기술원

전자전산학과 전산학전공

{haru*, kychwa}@jupiter.kaist.ac.kr

Improved Gossiping Algorithm in Rectangular Grids

Hyunseob Lee*, Kyung-Yong Chwa

Korea Advanced Institute of Science and Technology

Department of Electrical Engineering and Computer Science

Division of Computer Science

요 약

가쉬핑이란 그래프에서 각각의 점들이 각각 다른 메시지들을 다른 모든 점들에게 전달하는 것이다. 이 논문에서는 그리드 구조에서 각각의 점들이 매번 인접한 하나의 점과 하나의 메시지를 서로 주고받을 수 있는 조건 하에서 가쉬핑을 빠르게 하는 알고리즘을 제시한다. $m \times n$ 그리드에서 가쉬핑을 하는 기존의 알고리즘은 $(mn + \frac{1}{4}mn + 2)$ 라운드가 걸렸는데 이는 이 문제의 알려진 하한인 $(mn + 1)$과는 거리가 있었다. 우리는 mn이 홀수인 그리드 구조에서 가쉬핑을 $(mn + 9)$ 라운드 만에 끝낼 수 있는 알고리즘을 제시한다.

1 서론

'가쉬핑'은 그래프에서 각각의 점들이 갖고 있는 고유한 정보들을 다른 모든 점들에게 전달하는 과정이다. 초기에 모든 점들은 각각 고유한 패킷을 갖고 있고, 가쉬핑을 통해서 각각의 패킷들이 다른 모든 점들에게 전해지게 된다. 패킷은 그래프에서 인접한 점들 간의 패킷 교환을 통해서 전파된다.

가쉬핑은 원래 그래프 이론을 연구하는 수학자들이 처음 소개하였으나, 통신과 분산 환경에서의 시스템 등에 응용될 수 있다 [1].

이 논문에서는 패킷을 교환하는 점이 한 번에 하나의 패킷만을 주고 받을 수 있다고 가정한다. 이런 의미에서 우리는 '라운드' 개념을 쓴다. 한 라운드에 각 점은 오직 하나의 인접한 점과 패킷을 교환할 수 있다. 그리고 우리는 하나의 패킷은 한 라운드에 전달이 가능한 크기라고 가정한다(짧은 메시지의 가정 [2]). 일반적으로 이러한 가쉬핑 모델을 telephone model [3] 또는 full-duplex 1-port model [4]이라고 부른다.

$m \times n$ 그리드는 가로에 m개, 세로에 n개의 점이 격자 구조로 연결되어 있는 그래프를 말한다. 가로세로가 각각 m, n인 바둑판 모양이다. 지금부터는 $m \times n$ 그리드를 $G_{m,n}$으로 표시하겠다.

해밀토니안 그래프란 모든 점들을 한번만 지나는 사이클이 있는 그래프를 말하며 해밀토니안 사이클이라고도 부른다. 우리 모델에서 해밀토니안 그래프에서의 가쉬핑은 간단하면서도 라운드 수를 최소로 하는 최적의 알고리즘이 알려져있다 [1]. 그래서 일반적으로 그래프가 해

밀토니안 사이클을 갖고 있으면 가쉬핑을 빠르게 하는 최적의 알고리즘을 쉽게 만들 수 있다. $G_{m,n}$의 경우, mn이 짝수일 경우에는 해밀토니안 사이클을 갖게 되어서 최적의 가쉬핑 알고리즘을 얻을 수 있다 [1]. 하지만, mn이 홀수인 경우에는 최적의 알고리즘이 알려져있지 않고 이 문제의 하한은 $(mn + 1)$라운드로 [1] 알려져있고, 알려진 알고리즘 중 가장 빠른 알고리즘은 $(mn + \frac{1}{4}mn + 2)$라운드 [5]만에 가쉬핑을 끝낸다. 이 논문에서는 간단하면서도 개선된 가쉬핑 알고리즘을 제안하는데, mn이 홀수일 때 $(mn + 9)$ 라운드만에 가쉬핑을 끝낸다. 지금부터는 mn이 홀수라고 가정하자.

2 가쉬핑 알고리즘

일반적인 그래프 G에 대해서 가쉬핑의 하한은 다음과 같이 알려져 있다.

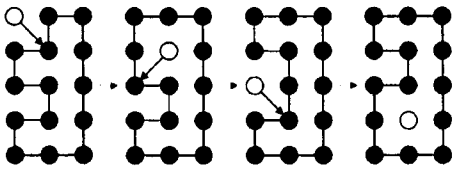
정 리 1 [1] *모든 그래프 $G = (V, E)$에 대해서, $N = |V|$ 일 때, G의 가쉬핑 시간의 하한을 $g(G)$라고 하면*

$$g(G) = \begin{cases} N-1 & N\text{이 짝수 일 때} \\ N & N\text{이 홀수 일 때} \end{cases}$$

이 정리는 해밀토니안 그래프에도 마찬가지로 적용되는데, 해밀토니안 그래프의 경우는 이 가쉬핑의 하한만큼 빠르게 하는 알고리즘이 알려져 있다 [1]. 해밀토니안 그래프에 최대한 큰 매칭을 잡은 다음에 매칭이 된 점들끼리 자신이 가진 패킷 중 가장 최근에 받은 패킷을 서로에게 준다. 그리고는 처음 잡은 매칭에 엇갈리게 다시 최대한 큰 매칭을 잡고서 역시 자신이 가진 패킷 중 가장 새로운 패킷을 서로에게 준다. 이렇게 두 가지 매칭을 번갈아 가면서 잡으면서 매번 서로 최근에 받은 패킷을 서로 주고받게 되면 각각의 패킷은 전체 사이클을 한 방향으로 돌면서 모든 점들을 다 방문하게 된다. 이 가쉬핑은 간단하면서도 빨라서, 약간 변형된 방법으로 우리의 가쉬핑 알고리즘에 이용을 한다.

$G_{m,n}$은 점이 홀수 개이기 때문에 최대한 큰 매칭을 잡더라도 항상 하나의 점은 매칭에서 제외가 되며 그 점은 패킷 교환에 참여할 수 없다. 이렇게 참여하지 못하는 점은 나중에 추가적인 라운드를 소비하여서 패킷을 보충을 받아야 하기 때문에 가쉬핑을 빨리 하는데 방해가 된다. 이렇게 매칭에서 제외가 되어 패킷 교환에 참여할 수 없게 되는 것을 이제부터 '쉬다'라고 표현하겠다. 한편 매 라운드마다 최소한 하나의 점은 반드시 쉬게 되는데 이 때 하나의 점이 여러 번 쉬는 것 보다는 여러 개의 점들이 나누어서 하나의 점이 쉬는 횟수를 최소한 하는 것이 나중에 추가적으로 필요한 라운드 수를 줄여서 전체적인 라운드 수를 줄이는데 유리하다. 다시 말하면 쉬는 점의 수를 최대화 하는 것이 가쉬핑을 빠르게 하는데 유리하다고 말할 수 있다.

우리 알고리즘의 기본적인 전략은 어쩔 수 없이 쉬는 점은 쉬게 하되, 여러 개의 점들이 나누어서 쉬고, 쉬는 점을 제외한 나머지 점들은 해밀토니안 사이클에서 가쉬핑을 하는 것처럼 패킷을 교환하게 하는 것이다. 여기서 중요한 것은 쉬는 점을 제외하고 나머지 점들은 사이클을 이루어서 패킷들은 해밀토니안 사이클에서 한 방향으로 모든 점들을 방문하는 것처럼 하는 것이다. 이를 위해서 우리는 쉬는 점들이 체계적인 순서에 따라 쉬고 그 순서에 따라 사이클이 조금씩 변하는 방법을 제안한다. 이 방법에서 쉬는 점들이 바뀔 때 마다 사이클이 바뀌기는 하지만 쉬는 점의 주변에서만 약간 바뀌고 사이클을 체계적으로 잘 바꾸기 때문에 전체적으로는 사이클이 변하지 않는 것처럼 작동한다.

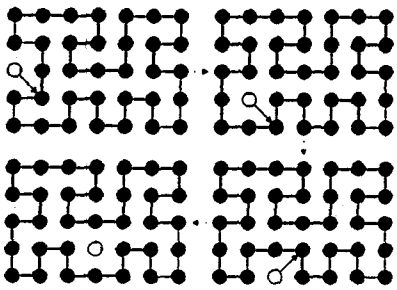


그 림 1: 쉬는 점이 전진하는 사이클의 변화

이제 우리는 위에서 말한 체계적인 사이클의 변화를 두 가지로 나누어서 보이겠다. 첫 번째는 그림 1에서 보는 바와 같이 쉬는 점이 지그재그로 한 방향으로 전진하는 순서로 쉬게 될 때 그림처럼 사이클을 만들어주게 되면, 우리가 원하는 대로 쉬지 않는 점들은 해밀토니안 사이클 상에서 가쉬핑을 하는 그대로 패킷을 교환할 수 있다. 그림에서 확인할 수 있다시피 쉬는 점이 바뀌는 부분을 제외하고는 나머지 사이클에는 변화가 없다. 이러한 방법으로 쉬는 점이 한쪽 끝에서 다른 한쪽 끝으로 돌아가면서 쉬게 할 수 있다. 여기에 쉬는 점을 더 늘이기 위해서 한쪽 끝에 도달하면 직각방향으로 방향을 바꾸어서 쉬는 점을 바꿀 수 있는 방법은 다음과 같다.

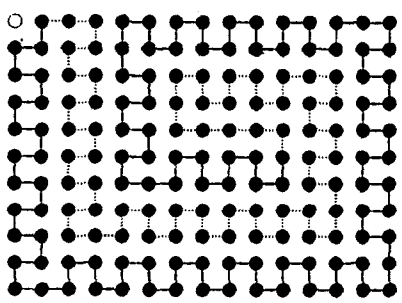
그림 2에서는 쉬는 점이 지그재그로 내려오다가 끝에 도달하기 한 칸 앞에서 다른 방향으로 옮겨가는 것을 볼 수 있다. 역시 쉬는 점들이 바뀌는 부분 외에는 사이클의 변화가 없음을 확인할 수 있다.

우리는 이 두 가지 방법을 이용하여 쉬는 점이 한 방향으로 옮겨가면서 쉬게 하다가 한쪽 끝에 다다르면 직각으로 방향을 바꾸어서 옮겨가면서 쉬게 할 수 있다. 이 두 가지 방법을 번갈아가면서 쓰면서 그리드를 바깥쪽부터 돌게 되면 전체적으로는 나선형을 그리면서 쉬는 점이 옮겨가면서 쉬게 할 수 있다. 이러한 사이클의 변화가 연속적으로 일어나게 하기 위해서는 초기 사이클을 그



그 림 2: 쉬는 점이 방향 바꾸는 사이클의 변화

림 3과 같이 만들어야 한다. 그림 3에서 처음 쉬는 점은 흰색 점으로 표시되어 있으며 이 쉬는 점은 실선이 전체적으로 나선형을 그리면서 이어져 가는 것을 따라서 옮겨가며 쉬게 된다. 알고리즘이 동작하면서 이 실선 부분이 조금씩 바뀌면서 쉬는 점이 옮겨 다니게 하며, 점선 부분은 변하지 않는 사이클 부분이 된다.



그 림 3: 초기 사이클

위에서 말한 대로 쉬는 점들이 쉬게 되면 $\frac{1}{6}mn$ 개의 점이 돌아가면서 쉬도록 할 수 있다. (정확한 분석은 생략하지만, 대략 $\frac{1}{4}mn$개의 점이 쉬게 되는 것을 그림에서 확인할 수 있다.) $\frac{1}{6}mn$ 개의 점이 각각 6번씩 쉬게 되면 대부분의 패킷들이 사이클을 따라서 대부분의 점들을 방문할 수 있게 된다. 각 패킷은 $mn-1$개의 점들을 방문해야 하므로 이 과정에서 $(mn-1)$라운드가 소비된다.

한편 쉬는 점에 대해서 생각해 보면, 쉬는 점은 쉬는 동안 패킷을 줄 수 없으며 받을 수도 없다. 그리고 쉬는 동안 쉬는 점을 우회하는 사이클을 지나가는 패킷도 받을 수 없다. 패킷의 입장에서 생각해보면 6번 쉬기 때문에 쉬는 동안 6개의 패킷을 놓친 셈이 된다. 따라서 추가적인 라운드를 소비해서 놓친 패킷을 받아야 한다. 그리고 쉬는 점이 갖고 있던 패킷에 대해서 생각해 보면, 이 패킷은 쉬는 점에 발이 묶여 있던 셈이 된다. 정상적으로 사이클을 돌아야 하지만 6번 동안 쉰 것이 된다. 이 패킷도 전체 사이클을 다 돌기 위해서 추가적인 라운드가 필요하다. 이렇게 놓친 패킷과 늦어진 패킷에 대한 처리를 하는데 10 라운드가 필요하다. 이렇게 해서 전체적으로는 $(mn+9)$ 라운드 만에 가쉬핑을 끝내게 된다. 뒤처리에 대한 구체적인 설명은 생략한다. 하지만 상수번의 라운드만을 더 소비한 다는 것은 명백하다. 패킷들이 쉬는 점을 우회해서 다른 점들을 방문할 때에 비록 쉬는 점은 방문하지 않지만, 쉬는 점을 제외한 나머지 쉬는 점의 모든 주변의 점들을 방문하기 때문에 쉬는 점이 놓친 패킷은 주변의 점에서 바로 받을 수 있다. 이 논문에서는 대부분의 패킷들이 $G_{m,n}$을 해밀토니안 그래프에서 가쉬핑을 하듯이 전달되는 방법을 제안한 것에 의의를 갖는다. 이로써 이전 논문에서 하한인 $(mn+1)$ 라운드에 대해서 $\frac{1}{4}mn$ 라운드를 더 필요로 하는 대신 상수 번의 라운드만을 더 필요로 하는 알고리즘을 가능하게 하였다.

3 결론

우리는 이 논문에서 $G_{m,n}$의 새로운 가쉬핑 알고리즘을 제안하였다. 이 알고리즘은 가쉬핑을 $(mn+9)$ 라운드 만에 끝내며 이전 결과인 $(mn+\frac{1}{4}mn+2)$ [5] 보다 개선된 결과이며 하한인 $(mn+1)$에 근접한 결과이다. 여전히 하한과의 거리는 있는 상태이며 최적의 가쉬핑 알고리즘을 찾는 연구가 필요하다.

참 고 문 헌

[1] Jean-Claude Bermond, Luisa Gargano, Adele A,Rescigno, Ugo Vaccaro. *Fast Gossiping by Short Messages*, SIAM Journal on Computing, 1998

[2] M. Soch, P. Tvrdik. *Time-optimal gossip of large packets in noncombining 2d tori an meshes.*, IEEE Trans. on Parallel and Distributed Systems, 10(12):1252-1261, 1999

[3] S. M. Hedetniemi, S. T. Hedetniemi, A. Liestman. *A survey of gossiping and broadcasting in communication networks*, Networks, 18:129-134, 1988

[4] D. W. Krumme, K. N. Venkataraman, G. Cybenko. *Gossiping in minimal time*, SIAM Journal on Computing, 21:111-139, 1992

[5] Jae-Hoon Kim, Jae-Ha Lee, Kyung-Yong Chwa. *Improved Gossipings by Short Messages in 2-Dimensional Meshes*, In Japan-Korea Joint Workshop on Algorithm and Computation, 2000