분산 트리거 계수 문제를 위한 간단하고 효율적인 알고리즘

이재흥
대전대학교 정보보안학과
e-mail : leejh@dju.kr

A Simple and Efficient Algorithm for the Distributed Trigger Counting Problem

Jaeheung Lee
Daejeon University, Department of Computer and Information Security

요 약

분산 트리거 계수 문제는 외부로부터 트리거를 수신하는 n개의 노드로 구성된 분산 시스템에서 수신한 전체 트리거 수의 합이 w에 이르렀을 때 이를 사용자에게 알려주는 문제로 다양한 분산 시스템 환경에서 모니터링과 전역 스냅샷을 위해 사용된다. 이 논문에서는 분산 트리거 계수 문제를 위한 간단하고 효율적인 알고리즘을 제안하고, 제안 알고리즘에서 수신한 전체 트리거 수의 합이 w에 이르렀을 때 이를 사용자에게 알려주지 않을 확률이 0임을 증명한다.

1. 서 론

분산 트리거 계수 문제(distributed trigger counting problem)는 외부로부터 트리거를 수신하는 n개의 노드로 구성된 분산 시스템에서 수신한 전체 트리거 수의 합이 w에 이르렀을 때 이를 사용자에게 알려주는 문제이다[1]. 트리거는 임의의 순서로 임의의 노드에서 수신되며 각 노드가 수신하는 트리거의 수 또한 정해져 있지 않다고 가정한다. 일반적으로 분산 트리거 계수 문제는 $w \gg n$을 가정하는데, 그렇지 않은 경우 각 노드가 트리거를 수신할 때마다 중앙 노드에게 메시지를 보내는 방식으로 $O(n)$개의 메시지만으로 문제를 해결하는 것이 가능하기 때문이다.

분산 트리거 계수 문제는 무선 센서 네트워크를 포함한 다양한 분산 시스템 환경에서 분산 모니터링과 전역 스냅샷을 위해 사용된다. 분산 모니터링은 교통량, 야생 동물 행동, 병력 이동 및 대기 조건과 같은 물리적 또는 환경적 조건을 모니터링하는 센서 네트워크에서 특히 중요하다. 또한 분산 시스템에서의 전역 스냅샷을 위해서는 모든 전송 중인 메시지를 기록할 필요가 있는데 Garg는 모든 전송 중인 메시지가 수신되었는지 여부를 판별하는 문제가 분산 트리거 계수 문제를 통해 해결될 수 있음을 증명하였다[2].

분산 트리거 계수 문제를 위한 알고리즘의 성능을 평가하기 위한 기준으로 메시지 복잡도와 MaxRcv가 있다[3]. 메시지 복잡도는 전체 노드가 주고받은 메시지 수의 총합을 의미하며, MaxRcv는 가장 많은 메시지를 수신한 노드가 받은 메시지 수를 의미한다. 메시지 복잡도가 전체 알고리즘의 효율성을 평가한다면 MaxRcv는 특정 노드에 부하가 가중되는지 여부를 평가하기 위해 사용된다.

이 논문에서는 분산 트리거 계수 문제를 위한 간단하고 효율적인 알고리즘을 제안한다. 또한 제안 알고리즘에서 수신한 전체 트리거 수의 합이 w에 이르렀을 때 이를 사용자에게 알려주지 않을 확률이 0임을 증명한다.

2. 관련 연구

분산 트리거 계수 문제를 효율적으로 해결하기 위한 다양한 알고리즘이 제안되었다[1-7]. 이 장에서는 그 중 대표적인 세 가지 알고리즘을 살펴본다.

2.1 Garg의 분산 트리거 계수 알고리즘들

Garg는 분산 시스템에서 전역 스냅샷을 위한 효율적인 알고리즘을 개발하면서 분산 트리거 계수 문제를 위한 세 가지 알고리즘을 제시하였다[2]. 이 세 가지 알고리즘은 그리드 기반, 트리 기반, 중앙 집중식 알고리즘으로 각각의 메시지 복잡도는 $O(n^{3/2})$, $O(n\log n\log(w/n))$, $O(n\log(w/n))$ 이며, MaxRcv는 각각 $O(\sqrt{n})$, $O(\log n\log(w/n))$, $O(n\log(w/n))$ 이다. 메시지 복잡도에 있어서는 중앙 집중식 알고리즘이 가장 효율적이지만 MaxRcv 관점에서는 트리 기반 알고리즘이 더 효율적이다. Garg는 또한 결정론적 분산 트리거 계수 알고리즘의 메시지 복잡도 하한이 $\Omega(n\log(w/n))$ 임을 증명하였다[2].

2.2 LayeredRand 알고리즘

전체 노드 수 n이 어떠한 정수 L에 대하여 $n=2^L-1$이라고 가정하자. LayeredRand 알고리즘에서 노드들은 이진트리와 비슷한 계층 구조를 형성한다[1]. 각 노드는 계층 0에서 $L-1$ 사이의 한 계층에만 존재하며, 계층 i에는 2^i개의 노드가 배치된다. ($2^0+2^1+\cdots+2^{L-1}=2^L-1$)

LayeredRand 알고리즘은 여러 라운드에 걸쳐 진행된다. 각 라운드가 시작될 때 아직까지 검출되지 않은 트리거의 수($\hat{w}$)를 알고 있어야 하며, 첫 라운드의 경우 $\hat{w}=w$이다. 각 노드 x는 지금까지 자신이 검출한 트리거의 수를 나타내는 변수 $C(x)$를 유지하며 자신이 속한 계층 l에 대하여 아래의 $\tau(l)$ 값을 계산한 후 $C(x)$가 $\tau(l)$에 도달하면 $C(x)$를 $\tau(l)$만큼 감소시키고 상위 계층의 노드 중 하나를 임의로 선택하여 코인 메시지를 전송한다.

$$\tau(l)=\left\lceil \frac{\hat{w}}{4\times 2^{l}\times\log(n+1)} \right\rceil$$

코인 메시지를 받은 계층 $l-1$의 노드 y는 $C(y)$를 $\tau(l)$만큼 증가시키고, $C(y)$가 $\tau(l-1)$에 도달했는지 확인한다. 만약 도달했으면 앞에서와 마찬가지로 $C(y)$를 $\tau(l-1)$만큼 감소시키고, 상위 계층의 노드 중 하나를 임의로 선택하여 코인 메시지를 전송한다. 마지막으로 루트 노드 r에 대해 $C(r)\geq\lceil\hat{w}/2\rceil$인 경우 전체 노드들이 수신한 트리거의 수를 집계하고 다음 라운드로 넘어간다.

LayeredRand 알고리즘의 메시지 복잡도는 $O(n\log n\log w)$ 이고, MaxRcv는 $O(\log n\log w)$ 이다.

2.3 CoinRand 알고리즘

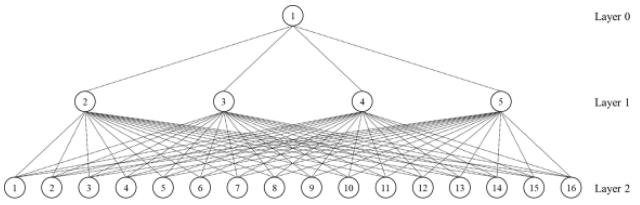
전체 노드 수 n이 어떠한 정수 L에 대하여 $n=2^{L}$이라고 가정하자. CoinRand 알고리즘은 0부터 L까지 $L+1$개의 계층으로 구성되며, 전체 n개의 노드가 계층 L의 리프 노드로 존재하면서 그 중 하나를 제외한 $n-1$개의 노드는 계층 0에서 $L-1$ 사이의 한 계층에도 같이 포함된다[3].

CoinRand 알고리즘 역시 라운드 기반으로 동작한다. 각 라운드가 시작될 때 임계값 $\tau=\lceil\hat{w}/4n\rceil$를 계산하고, 자신이 검출한 트리거의 수를 나타내는 변수 $C(x)$가 임계값 τ에 다다르면 $C(x)$를 τ만큼 감소시키고 계층 $L-1$에 속한 임의의 노드를 선택하여 코인 메시지를 전송한다. 코인 메시지를 받은 내부 노드 y는 처음 코인 메시지를 받았을 때는 이를 간직할 뿐 특별한 추가 동작을 하지 않지만 두 번째부터는 자신의 상위 계층에 속한 임의의 노드를 선택하여 코인 메시지를 전송한다. 마지막으로 루트 노드가 코인 메시지를 받으면 전체 노드들이 수신한 트리거의 수를 집계하고 다음 라운드로 넘어간다.

CoinRand 알고리즘의 메시지 복잡도와 MaxRcv는 각각 $O(n(\log w+\log n))$, $O(\log w+\log n)$ 이다.

3. 제안 기법

전체 노드 수 n이 어떠한 정수 k에 대하여 $n=k^{2}$이라 가정하자. 제안 알고리즘은 계층 0부터 2까지의 세 계층으로 구성된다. CoinRand에서와 마찬가지로 전체 n개의 노드가 계층 2의 리프 노드로 존재하면서 그 중 하나의 노드는 계층 0에, k개의 노드는 계층 1에 같이 포함된다. 계층 0과 계층 1에 있는 $k+1$개의 노드는 전체 노드가 라운드로빈 방식으로 돌아가며 담당한다. 그림 1은 $k=4$일 때의 계층 구성을 보여준다.



(그림 1) 제안 알고리즘 계층 구성 ($k=4$)

제안 알고리즘 역시 라운드 기반으로 동작한다. 각 라운드가 시작될 때 임계값 $\tau_{leaf}=\lfloor\hat{w}/2n\rfloor$를 계산하고, 자신이 검출한 트리거의 수를 나타내는 변수 $C(x)$가 임계값 τ_{leaf}에 다다르면 $C(x)$를 τ_{leaf}만큼 감소시키고 계층 1에 속한 임의의 노드를 선택하여 코인 메시지를 전송한다. 코인 메시지를 받은 노드 y는 자신이 받은 코인 메시지의 수를 나타내는 변수 $D(y)$ 값을 1 증가시키며 이 값이 또 다른 임계값 $\tau_{internal}=\lfloor k/2\rfloor$와 같거나 큰지 확인한다. 만약 같거나 큰 경우 $D(y)$ 값을 $\tau_{internal}$만큼 감소시키고 루트 노드에게 코인 메시지를 전송한다. 마지막으로 루트 노드는 자신이 받은 코인의 개수가 k개에 이르면 전체 노드들이 수신한 트리거의 수를 집계한 후 다음 라운드로 넘어간다.

라운드가 진행되면서 $\hat{w}<2n$가 되면 최종 라운드가 시작된다. 최종 라운드에서 각 노드는 발생한 모든 트리거 정보를 루트 노드에게 직접 보내고 루트 노드는 $\hat{w}$가 0이 되었을 때 이를 사용자에게 알려주면서 알고리즘이 종료된다.

아래 정리 1과 정리 2는 제안 알고리즘에서 수신한 전체 트리거 수의 합이 w에 이르렀을 때 이를 사용자에게 알려주지 않을 확률이 0임을 증명한다.

정리 1. 제안 알고리즘의 모든 라운드에서 $\hat{w}\geq 2n$인 경우 각 노드는 $\tau_{leaf}=\lfloor\hat{w}/2n\rfloor$개의 트리거를 받을 때마다 하나의 코인 메시지를 계층 1의 임의의 노드에게 전송한다. 이 경우 해당 라운드의 트리거 분포와 관계없이 항상 최소한 n개의 코인 메시지가 보내진다.

증명. $\hat{w}\geq 2n$인 경우 각 노드가 코인 메시지를 발생시키지 않고 받을 수 있는 트리거의 수는 $\hat{w}/2n-1$보다 작거나 같다. 따라서 n개의 코인 메시지가 보내졌을 때 발생 가능한 트리거의 수는 $(\hat{w}/2n)\times n+(\hat{w}/2n-1)\times n<\hat{w}$을 통해 $\hat{w}$보다 작은 것을 알 수 있다.

정리 2. 제안 알고리즘의 모든 라운드에서 $\hat{w}\geq 2n$인 경우 계층 1에 있는 k개의 노드는 $\tau_{internal}=\lfloor k/2\rfloor$개의 코인 메시지를 받을 때마다 하나의 코인 메시지를 루트 노드에게 전송한다. 이 경우 해당 라운드의 코인 분포와

관계없이 항상 최소한 k개의 코인 메시지가 루트 노드에게 보내진다.

증명. $\hat{w} \geq 2n$인 경우 정리 1에 의해 계층 1에 있는 k개의 노드는 최대 $n=k^2$개의 코인 메시지를 받는다. 계층 1에 있는 노드가 코인 메시지를 발생시키지 않고 받을 수 있는 코인 메시지의 수는 $k/2-1$보다 작거나 같다. 따라서 k개의 코인 메시지가 보내졌을 때 발생 가능한 코인 메시지의 수는 $(k/2)\times k+(k/2-1)\times k < k^2 = n$을 통해 n보다 작은 것을 알 수 있다.

4. 결론

이 논문에서는 분산 트리거 계수 문제를 위한 간단하고 효율적인 알고리즘을 제안하였다. 또한 제안 알고리즘에서 수신한 전체 트리거 수의 합이 w에 이르렀을 때 이를 사용자에게 알려주지 않을 확률이 0임을 증명하였다. 향후 추가 연구로 다양한 분석 및 실험을 통해 제안 알고리즘의 성능을 평가할 예정이다.

참 고 문 헌

[1] V. T. Chakaravarthy, A. R. Choudhury, V. K. Garg, and Y. Sabharwal, “Efficient Decentralized Algorithms for the Distributed Trigger Counting Problem,” Theory Computing Systems, vol. 51, no. 4, pp. 447 - 473, 2012, doi: 10.1007/s00224-012-9405-4.

[2] R. Garg, V. K. Garg, and Y. Sabharwal, “Efficient Algorithms for Global Snapshots in Large Distributed Systems,” IEEE Transactions on Parallel Distributed Systems, vol. 21, no. 5, pp. 620 - 630, 2010, doi: 10.1109/TPDS.2009.108.

[3] V. T. Chakaravarthy, A. R. Choudhury, and Y. Sabharwal, “Improved Algorithms for the Distributed Trigger Counting Problem,” in 2011 IEEE International Parallel & Distributed Processing Symposium, 2011, pp. 515 - 523, doi: 10.1109/IPDPS.2011.56.

[4] S. Kim, J. Lee, Y. Park, and Y. Cho, “An optimal distributed trigger counting algorithm for large-scale networked systems,” Simulation, vol. 89, pp. 846 - 859, 2013, doi: 10.1177/0037549713485499.

[5] Y. Emek and A. Korman, “Efficient Threshold Detection in a Distributed Environment: Extended Abstract,” in Proceedings of the 29th ACM SIGACT-SIGOPS Symposium on Principles of Distributed Computing, 2010, pp. 183 - 191, doi: 10.1145/1835698.1835742.

[6] C.-C. Chang and J. Tsai, “Distributed Trigger Counting Algorithms for Arbitrary Network Topology,” Wireless Communication Mobile Computing, vol. 16, no. 16, pp. 2463 - 2476, 2016, doi: 10.1002/wcm.2698.

[7] S. Kim and Y. Park, “DDR-coin: An Efficient Probabilistic Distributed Trigger Counting Algorithm,” Sensors, vol. 20, no. 22, 2020, doi: 10.3390/s20226446.