포아송·감마 분포를 활용한 컨테이너 기반 병렬 해시 탐색의 대기시간 및 오버헤드 실증 분석

유민수[1], 김건민[2], 김경백[2]
[1]전남대학교 빅데이터융합학과
[2]전남대학교 인공지능융합학과

kakaominsus@gmail.com, geonminkim@jnu.ac.kr, kyungbaekkim@jnu.ac.kr

Empirical Analysis of Waiting Time and Overhead in Container-Based Hash Search using Poisson and Gamma Distributions

Minsoo Yoo[1], Geonmin Kim[2], Kyungbaek Kim[2]
[1]Dept. of Big Data Convergence, Chonnam National University
[2]Dept. of Artificial Intelligence Convergence, Chonnam National University

요 약

본 논문은 분산 병렬 탐색 알고리즘의 평균 시간 복잡도를 포아송 과정과 감마 분포를 활용하여 수학적으로 유도하고, Docker Compose 환경에서 실증적으로 검증한다. N개의 독립 워커가 빈도 λ의 포아송 과정으로 해시 프리픽스 매칭을 수행할 때, k개 부분 해 수집까지의 대기시간은 감마 분포를 따르는 것이라고 증명한다. N=1,2,4,8,16 워커에 대해 각 30회 반복 실험을 수행한 결과, N≤4에서는 이론값과 실측값의 오차가 3% 미만으로 모델의 유효성을 확인하였다. 반면 N≥8에서는 오버헤드가 비선형으로 급증하여 이론적 스케일아웃의 한계를 정량화하였으며, 단일 호스트 환경에서의 실용적 최적 노드 수가 4~8 범위임을 도출하였다.

1. 서론

병렬 탐색 알고리즘의 성능 분석은 전통적으로 최악-경우 시간 복잡도에 의존해왔다[1]. 그러나 이 접근법은 해가 탐색 공간의 끝에 위치한다고 가정하여 실제 수행 시간을 비관적으로 평가한다. 분산 병렬 탐색 환경에서는 각 노드의 탐색 성공이 확률 과정으로 모델링되어야 하며, 평균-경우 분석이 더 현실적인 성능 지표를 제공한다[2]. 이론적으로 N개의 독립 워커가 포아송 과정을 따를 때, k개의 부분 해를 수집하는 대기시간의 기대값은 E[T]=k/Nλ로 유도된다. 그러나 실제 컨테이너 기반 분산 환경에서는 IPC(Inter-Process Communication) 비용과 CPU 컨텍스트 스위칭 등의 오버헤드로 인해 이론값과 실측값 사이에 차이가 발생한다[3]. 확률적 병렬 탐색의 이론적 기반[2]과 분산 컴퓨팅의 협력 오버헤드는 선행연구에서 개별적으로 다루어졌으나, 컨테이너 환경 실증을 통해 이 괴리를 통합적으로 정량화한 연구는 존재하지 않는다. 이에 본 연구는 실제 컨테이너 환경에서 발생하는 오버헤드를 정량화하고, 성능이 포화되는 최적 임계점을 도출한다. 본 논문의 기여는 다음과 같다: (C1) 포아송/감마 분포 기반 분산 병렬 탐색의 평균 시간 복잡도를 수학적으로 유도하고, (C2) Docker Compose 환경에서 이론-실측 차이를 정량화하며, (C3) 오버헤드 보정 모델을 도출하여 실용적 최적 노드 수를 제시한다.

2. 수학적 모델: 포아송 및 감마 분포를 활용한 성능 유도

2.1 SHA-256 해시 프리픽스 매칭 확률적 모델링

단일 워커가 SHA-256 해시 프리픽스 매칭을 수행할 때, 각 해시 시도는 독립이며 성공 확률 p=1/16^d (d: 16진수 프리픽스의 0 개수)로 일정하다. 본 실험에서는 d=6(성공 확률 p≈5.96×10^(-8))으로 설정하였다. 단위시간당 성공 횟수는 빈도 λ의 포아송 과정을 따르며, λ = (초당 해시 수)× p로 N=1 단일 워커 실측에서 산출된다. SHA-256 병렬화의 성능 특성은 소프트웨어 및 하드웨어 관점에서 연구된 바 있으나, 확률 모델 기반 평균 대기시간 분석과 결합한 연구는 부재하다[5].

2.2 감마 분포를 통한 이론적 평균 대기시간 산출

N개의 독립 워커가 각각 빈도 λ의 포아송 과정을

따르면, 포아송 과정의 중첩 정리에 의해 전체 시스템의 합산 과정은 빈도 Nλ의 포아송 과정이 된다. k번째 사건까지의 대기시간 T는 Gamma(k,Nλ) 분포를 따르며 그 기대값은 수식 (1) 과 같다.

$$E[T] = k/N\lambda \quad (1)$$

이는 워커 수 N에 반비례하는 이상적인 선형 스케일아웃을 예측한다. 실제 환경에서는 동기화 오버헤드가 존재하므로, 실측 대기 시간을 수식 (2)와 같이 보정한다.

$$E[T_{real}] = k/N\lambda + \delta(n) \quad (2)$$

여기서 오버헤드는 N개의 워커 간의 IPC 및 CPU 컨텍스트 스위칭에 의한 오버헤드이다.

3. 실험 설계: Docker 기반 병렬 해시 탐색

3.1 하드웨어 사양 및 Docker 컨테이너 환경

본 실험은 단일 호스트(CPU: AMD Ryzen 5 6500F 6-Core, RAM: 32GB)에서 Docker Compose를 활용하여 N=1,2,4,8,16개의 워커 컨테이너를 구성하였다. 컨테이너 기반 워크로드의 런타임 성능 특성은 [4]에서 분석된 바 있으며, 본 연구는 이를 병렬 탐색 시나리오에 적용한다. 각 워커는 Python 3.11 환경에서 SHA-256 해시 프리픽스 매칭을 수행하며, DIFFICULTY=6으로 설정하였다. 특히, 각 N 조건(1, 2, 4, 8, 16)에 대해 k=10(부분 해 개수)으로 설정하고 30회 반복하여 총 150회의 실험을 수행하였다. DIFFICULTY=6은 단일 워커 기준 약 90~120초가 소요되도록 사전 캘리브레이션하여 오버헤드의 통계적 유의성을 확보하였다.

3.2 워커 독립성 보장 및 기준 성능 λ 측정

워커 간 탐색 공간의 독립성을 보장하기 위해, 각 컨테이너에 고유 ID 기반 난수 시드를 부여하였다. 이를 통해 워커 간 탐색 영역의 중복을 방지하고, 포아송 가정의 독립성 조건을 충족하였다. 기준 탐색 속도 λ는 N=1 단일 워커의 30회 반복 실측이다.

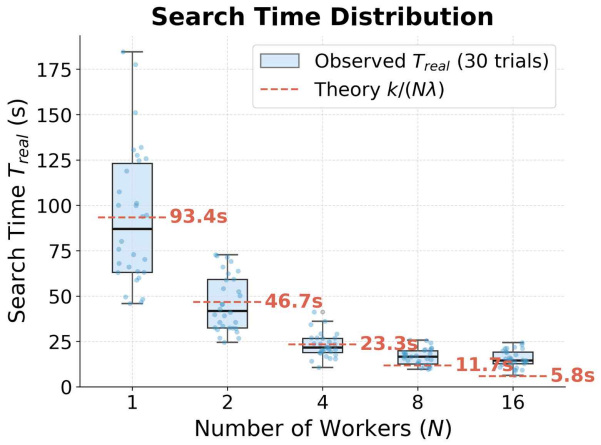
4. 실험 결과 분석: 이론 모델 검증 및 오버헤드 정량화

4.1 소규모 병렬화에서 이론-실측값 일치도 검증

N=2에서 스피드업 2.03배, N=4에서 4.08배로 이론값과의 오차가 3% 미만이며, E[T] 모델이 소규모 병렬화에서 높은 정확도를 보임을 확인하였다. 실측 표준편차는 모든 N 조건에서 이론 예측보다 다소 크게 나타났는데, 이는 포아송 근사의 한계로 시스템 수준의 미세 변동이 추가 분산으로 작용한 것으로 해석된다. 오버헤드가 음수인 것은 통계적 변동 범위 내에서 이론보다 빠르게 수행됨을 의미한다.

그림 1은 각 N 조건에서 30회 반복 실험의 탐색 시간 분포와 이론-실측 비교를 하나의 그래프에 나타낸 것이다. N≤4에서는 이론 예측값(빨간 점선)이 박스플롯의 중앙값 및 실측 평균(파란 원) 근처에 위치하며, 95% 신뢰구간 내에 이론값이 포함되어 모델의 유효성을 확인할 수 있다. 반면 N=8부터 이론값이 박스플롯 하단 아래로 벗어나고 실측 평균과의 괴리가 급격히 확대되어, 오버헤드를 정량적으로 확인할 수 있다. 또한 N 증가에 따른 사분위 범위 감소는 포아송 중첩의 분산 축소 효과와 일치한다.

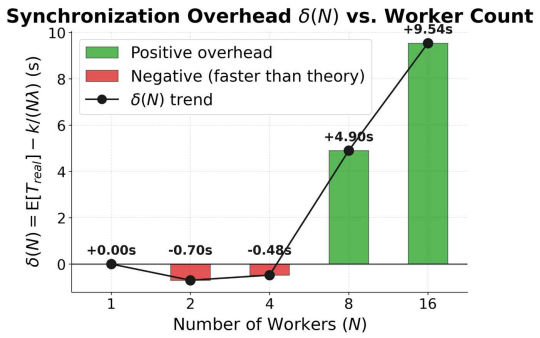
4.2 물리 코어 초과에 따른 비선형적 오버헤드 $\delta(n)$ 분석



(그림 1) 워커 수 N 별 탐색 시간 분포 및 이론-실측 비교(k = 10, λ = 0.1071, 각 30회 반복).

<표 1> 각 N에 대한 평균 실측시간, 이론 예측 시간, 오버헤드, 스피드업

N	E[T] 실측	SD	95% CI	E[T] 이론	오버헤드 (s)	스피드업
1	93.38	38.10	[79.15, 107.61]	93.38	0.00	1.00
2	45.99	15.49	[40.21, 51.78]	46.69	-0.70	2.03
4	22.87	6.67	[20.38, 25.36]	23.35	-0.48	3.08
8	16.57	4.54	[14.88, 18.27]	11.67	4.90	5.64
16	15.37	4.61	[13.65, 17.09]	5.84	9.54	6.08



(그림 2) 워커 수 N에 따른 동기화 오버헤드

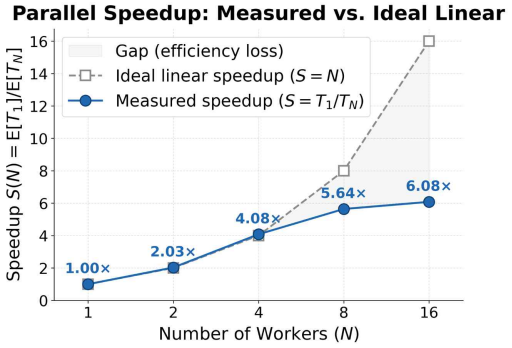
<표 2> 워커 분배 및 병렬 효율 분석

N	코어 당 워커	스레드 당 워커	$\delta(n)$	병렬 효율(%)
1	0.17	0.08	0.00	100.0
2	0.33	0.17	-0.70	101.5
4	0.67	0.33	-0.48	102.1
8	**1.33**	0.67	4.90	**70.4**
16	**2.67**	1.33	9.54	**38.0**

표 2와 같이 N=8부터 오버헤드가 양수로 전환되어 급증한다. N이 8일 때 4.90초, N이 16일 때 9.54초로, 이론적으로 기대되는 스피드업에 비해 실제 스피드업은 5.64배, 6.08배에 그쳤다. 그림 2는 오버헤드의 변화 패턴을 시각화한 것으로, N≤4에서 오버헤드가 음수로 나타나는 것은 통계적 변동 범위이며, N=8부터 양수로 전환되어 비선형적으로 급증한다. 이는 호스트 CPU의 물리 코어 수(6코어)를 초과하는 시점으로, 컨텍스트 스위칭 비용이 지배적 오버헤드로 작용함을 보여준다. 또한, N=8부터 병렬 효율이 70.4%, N=16에서는 38.0%로 감소한다.

4.3 자원 효율성을 고려한 실용적 최적 노드 수

그림 3과 같이 N=4→8 구간에서 추가 스피드업이 1.38배에 불과하며, N=8→16 구간에서는 1.08배로 포화 상태이다. 따라서 단일 호스트 Docker 환경에서의 실용적 최적 노드 수는 N=4~8 범위로 이는 오버헤드 대비 성능 향상의 효율이 급격히 감소하는 변곡점에 해당한다.



(그림 3) 이상적 스피드업(y=N)과 실측 스피드업 비교

5. 결론 및 향후 연구

본 연구는 포아송 및 감마 분포 기반으로 병렬 탐색의 평균 수행 시간을 이론적으로 유도하고, 컨테이너 환경에서 이를 검증하였다. 그 결과, 물리 코어 수를 초과하지 않는 구간에서는 이론 모델과 실측값이 높은 일치도를 보이며, 확률적 평균 시간 모델의 유효성을 확인하였다. 반면, 워커 수가 물리 코어 수를 초과하는 구간에서는 IPC 및 컨텍스트 스위칭 오버헤드로 인해 성능이 비선형적으로 저하되었으며, 이론값과 실측값 간의 차이로 실제 컨테이너 환경에서는 단순한 선형 스케일링 가정이 성립하지 않음을 확인하였다. 향후 Kubernetes 기반 멀티 노드 환경으로 확장하여 네트워크 지연을 포함한 오버헤드 모델을 분석하고, 동적 워커 조정을 통한 적응형 병렬 탐색 기법으로 확장할 계획이다.

Acknowledgement

본 연구는 2025년도 과학기술정보통신부 및 정보통신기획평가원의 소프트웨어중심대학사업의 연구결과로 수행되었습니다.(2021-0-01409)(50%)본 연구는 과학기술정보통신부 및 정보통신기획평가원의 인공지능융합혁신인재양성사업 연구 결과로 수행되었음(IITP-2026-RS-2023-00256629)(50%)

참고문헌

[1] M. Mitzenmacher and E. Upfal, "Probability and Computing: Randomized Algorithms and Probabilistic Analysis," Cambridge University Press, 2005.

[2] R. Karp and Y. Zhang, "Randomized parallel algorithms for backtrack search and branch-and-bound computation," Journal of the ACM, Vol.40, No.3, pp.765-789, 1993.

[3] J. Dean and S. Ghemawat, "MapReduce: Simplified Data Processing on Large Clusters," OSDI, San Francisco, 2004, pp.137-150.

[4] J. Smith et al., "Performance Overhead of Containerized Applications in Distributed Environments," IEEE Access, vol. 10, pp. 12345-12356, 2022.

[5] Qianyun Wang, "A chaotic parallel hash engine with dynamic stochastic diffusion for blockchain and cloud security," Scientific Reports, Vol.15, No.37945, 2025