병렬분산컴퓨팅 (CSE5414)

Fall 2020

Assignment #2

이름: 김인호

학번: 20161577

담당 교수: 박성용 교수님

20161577
김인호

병렬분산컴퓨팅 (CSE5414)

Assignment #2

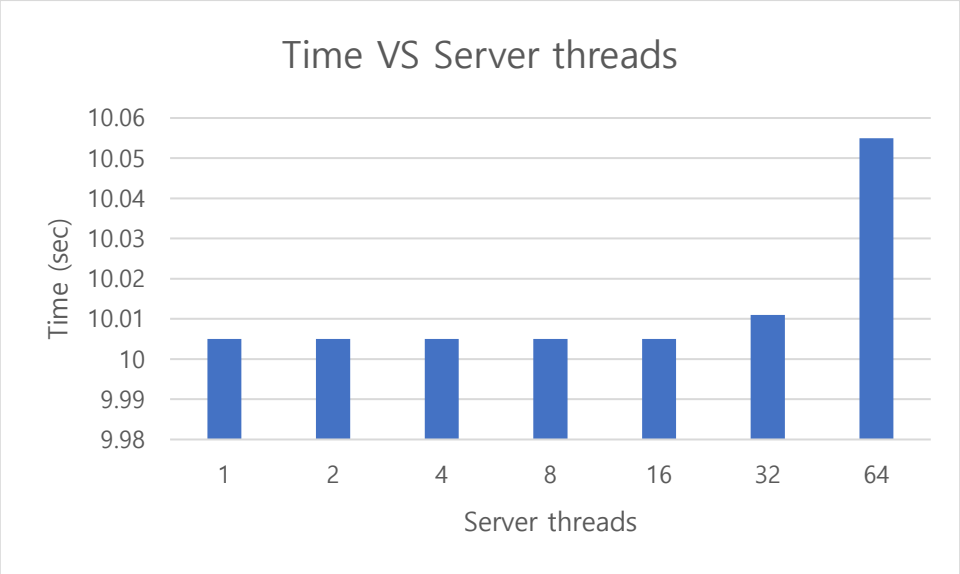
1. Server thread 개수에 따른 변화

Client thread 개수: 4 개

Client thread 당 request 개수: 10 개

Request 사이의 interval: 1 초

Server threads	1	2	4	8	16	32	64
Time (sec)	10.005	10.005	10.005	10.005	10.005	10.011	10.055



Server 에서 request 를 처리하기 위한 thread pool 의 크기를 변화시켰을 때의 결과이다. 미세한 차이이지만 thread 개수가 증가할수록 thread 간의 동기화 등의 overhead 가 커져서 처리 시간이 약간 증가하는 것을 볼 수 있다.

2. Client thread 개수에 따른 변화

Server thread 개수: 8개

Client thread 당 request 개수: 3개

Request 사이의 interval: 1초

Client threads	1	2	4	8	16	32
Time (sec)	10.006	10.006	10.004	10.007	10.008	10.010

Client에서 request를 보내는 thread 개수를 변화시켰을 때의 결과이다. 위 경우와 마찬가지로 미세한 차이이지만 thread 개수가 증가할수록 overhead가 커져서 처리 시간이 약간 증가하는 것을 볼 수 있다.

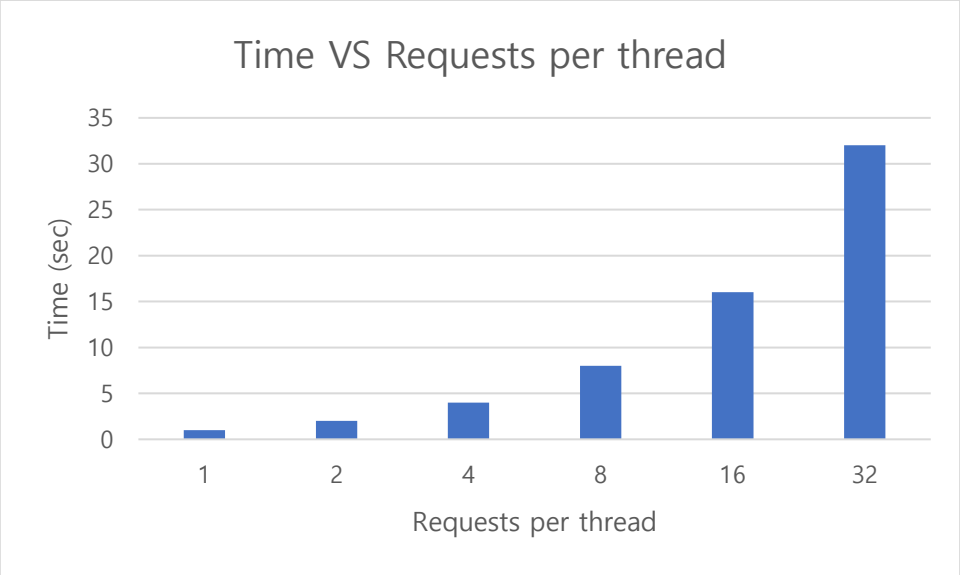
3. Request 개수에 따른 변화

Client thread 개수: 2개

Server thread 개수: 8개

Request 사이의 interval: 1초

Req. per thread	1	2	4	8	16	32
Time (sec)	1.001	2.002	4.003	8.004	16.007	32.013



Client에서 request를 보내는 thread 각각이 보내는 request의 개수를 변화시켰을 때의 결과이다. 시간이 request의 총 개수에 비례하며 증가하는 것을 볼 수 있지만 이것은 사실 request 사이의 interval이 1초로 설정되어 있어서 그런 것이다. 대략적으로 각 경우마다 총 request 개수 * 1초를 뺀 값(소수점 부분)을 보면 0.001초, 0.002초, 0.003초, 0.004초, 0.007초 그리고 0.013초로 증가한다. 총 request 개수가 8개인 경우 0.003초이고 64개인 경우 0.013초인 것을 살펴보면 총 request 개수는 8배로 증가했지만 처리시간은 약 4배로 증가했다. 이것으로 multithreaded server의 효과를 볼 수 있다.

4. Cache의 영향

Cache 작용은 분명 있었을 것이다. 위 결과에서 볼 수 있듯이 thread 개수나 request 개수에 따른 큰 차이는 관찰할 수 없었다. 실행 환경에 따라 결과가 다르겠지만 큰 용량의 파일에 대한 request 경우에도 큰 차이를 관찰할 수 없었다.

5. Epoll을 활용한 peer model

Epoll을 활용하면 커널 level에서 file descriptor를 관리하기 때문에 CPU가 모든 file descriptor의 변화를 매번 확인할 필요가 없다. 어떤 file descriptor에서 이벤트가 발생한다면 해당 file descriptor만 넘겨주기 때문에 CPU time이 줄어들 수 있고 이를 활용한 server의 경우에도 더욱 빨리 request에 대한 처리를 할 수 있다.