**기말고사 답안**

1. 1)loop carried dependency를 제거하여 parallelsim을 얻을 수 있다.

또는 2)루프 iteration에 필요한 branch 등의 인스트럭션 수가 감소한다.

2.  TCP는 3-way handshake를 통해 연결을 형성하여 데이터를 송수신하는 연결지향형 프로토콜이며, 데이터 유실시 데이터 재전송을 요청하는 등 신뢰성을 보장한다. UDP는 호스트가 up인지 상관하지 않고 데이터를 송신만 하는 비연결지향형 프로토콜이다. 그렇기 때문에 데이터 유실의 가능성이 있지만 streaming service 등에서는 TCP에 비해 좋은 성능을 보일 수 있다.

3. 예를 들어, 소켓 통신을 하는 서버가 있을 때, 클라이언트 당 하나의 프로세스로 detach해서 connection을 유지하게 되면, 서로 다른 프로세스이므로 IPC를 사용해 복잡한 communicaiton을 할 수 밖에 없으며 connection에 대한 정보보다 훨씬 큰 다른 데이터들도 공간을 차지하게 된다. I/O multiplexing을 사용하면 하나의 socket fd로부터 여러 클라이언트의 write를 감지할 수 있어 효율적인 자원 사용이 가능하다.

4. 1)사용하려는 데이터 간의 dependency가 존재하면, 서로 상대의 데이터를 기다리면서 더 낮은 성능을 보일 수도 있을 것이다.

또는 2)두 쓰레드가 사용하는 공유 자원(메모리 등)의 양이 전체 자원의 양을 넘어 두 쓰레드가 경쟁하여 자원을 사용할 때 이런 현상이 발생할 수 있다. 예를 들어, 다른 쓰레드가 사용하고 있는 메모리 공간을 evict시켜 오버헤드가 발생할 수도 있다. 

5. 정의 : 여러 thread가 한 캐시 블락을 공유할 때, 쓰레드들이 해당 블락의 서로 다른 데이터에 쓰기를 하여 실제로는 겹치지 않지만 서로 invalidation을 계속하는 상황.

해결 방법 : 패딩을 추가하여 각 쓰레드의 데이터가 같은 캐시 블락에 들어가지 않도록 하여 해결할 수 있다.

6. 정의 : 한 사이클에 여러 인스트럭션을 이슈하고 실행할 수 있는 프로세서.

성능 향상 : 대부분의 프로그램이 가진 Instruction Level Parallelism을 이용하여 성능을 높일 수 있다.

7. [I,j,k]를 사용할 경우 연속된 메모리 공간을 사용하기 때문에 접근시 캐시 미스가 최소화 된다. 그러나 [jki] [kij]를 사용하면, 연속되지 않은 공간에 접근하게 되어 하나의 캐시 블락에 있을 확률이 작아져서 캐시 미스 레이트가 증가한다. 즉, 낮은 spatial locality로 인해 cache miss rate이 높아져 메모리 접근 빈도가 증가하게 되어 성능이 떨어진다.

8. 캐시가 존재하지 않고 메모리만 사용하는 컴퓨터라는 것을 알 수 있다.  
또는 4GB 이상의 캐시를 사용한다는 점을 알 수 있다.

중간2 답안

1. Class A/B/C와 같이 private addressing을 사용하는(ex. 172.16.0.0/12) NAT나 하나의 public address에서 여러 포트를 다른 host들과 매핑하여 사용하는 PAT를 활용할 수 있다.

2. header = 32bit, 이 중 마지막 2 bit는 항상 0으로 사용하므로 size에 사용가능한 비트 수 30비트이다.

따라서 (2^30 - 1) \* 4  = (2^32 - 4) Byte 사용 가능

3. free block으로 읽기 위해서는 블락 앞과 뒤의 32bit가 동일하여야 하며 마지막 두 비트가 00이어야 한다.

따라서, 앞과 뒤 30비트가 같고, 끝이 00인 sequence 개수 = 2^30 / 모든 두 32bit sequence의 수 = 2^64개의 확률이다.

즉, 2^30/2^64 = 1/2^34.

4. f1의 경우는 integer pointer 변수를 리턴하는 함수이다. f2는 함수의 포인터로써 해당 함수는 integer 변수를 반환한다.

5. Listening descriptor: 클라이언트 연결 요청의 end point, 서버 당 한 번만 생성됨

Connected descriptor: 클라이언트와 서버 사이의 연결의 end point, 서버가 클라이언트로부터의 연결 요청을 accept할 때마다 생성됨. 해당 클라이언트를 서비스하는 동안 유지

이 둘을 구분하는 것은 port multiplexing 이용.

6.양 끝의 소켓 주소, 즉 ip 주소와 port의 쌍으로 유일하게 결정되는 인터넷 호스트 간의 연결. 서버와 클라이언트는 바이트의 스트림을 connection을 통해 주고받으며 통신함.

7. Zero length chunk를 보내는 것으로 마지막 chunk임을 표시.

8. web server가 IPC, shared memory, 환경 변수를 통해 자식 프로세스에 전달할 수 있다.