Effective Java 읽기 스터디 10min Mini Seminar

5주차 2021.03.29

메모리 관리 전략

발표자: 김정수

Contents

1. 연속 메모리 할당 방식
2. 단편화
3. 가상 메모리
4. 페이지 교체 알고리즘

1. 연속 메모리 할당 방식

프로세스별로 메모리 공간을 독립적으로 가지고 있게 해야 한다.
⇒ 메모리 공간이 공유될 경우 동시에 실행될 때 문제 발생
⇒ 한 번에 많은 프로그램을 실행시키려면 메모리 관리가 중요하다.

Dynamic Loading
: 모든 루틴은 디스크에 있다가 필요할 때만 메모리로 적재된다.
⇒ 반복 호출이 잘 일어나지 않는 프로그램일 때 유용함

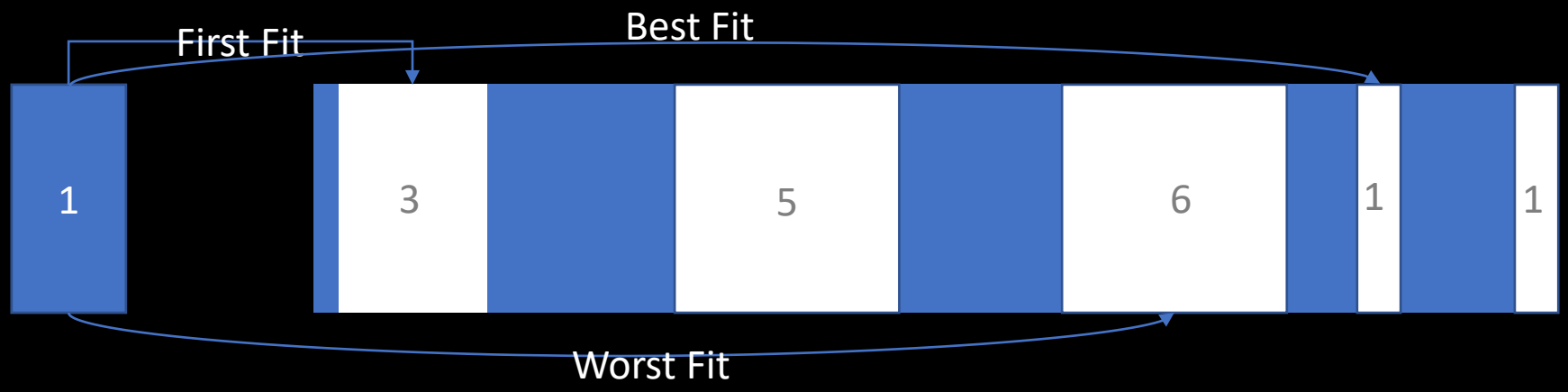
Dynamic Linking
: 프로그램이 동작할 때 라이브러리 루틴을 연결한다.
⇒ stub이 라이브러리 루틴을 어떻게 불러오는지 알려준다.

⇒ 연속 메모리 할당에서는 한 프로세스가 하나의 연속된 메모리 공간을 가지게 된다.

1. 연속 메모리 할당 방식

시스템은 새로운 프로세스에 할당될 메모리를 배치하기 위해 3가지 방법을 사용한다.

1) First fit
   - 사용 가능한 hole을 발견하는 즉시 탐색을 종료한다.
2) Best fit
   - 메모리 할당이 일어났을 때 가장 작은 hole이 생기는 곳을 찾는다.
3) Worst fit
   - 가장 큰 hole을 찾는다. 생기는 hole에 다른 프로세스를 할당할 수 있다면 best-fit보다 유용할 수 있다.

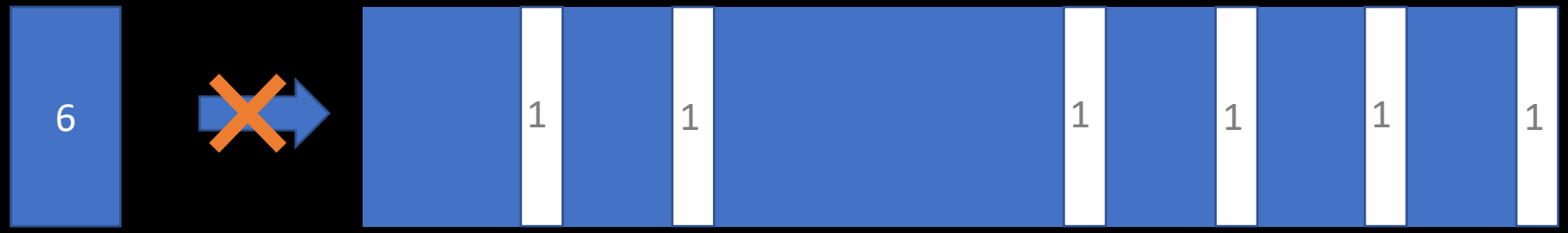


=> 메모리 할당과 제거를 반복하다 보면 hole이 많이 생기게 되고 외부 단편화가 발생한다.

2. 단편화 Fragmentation

1) 외부 단편화

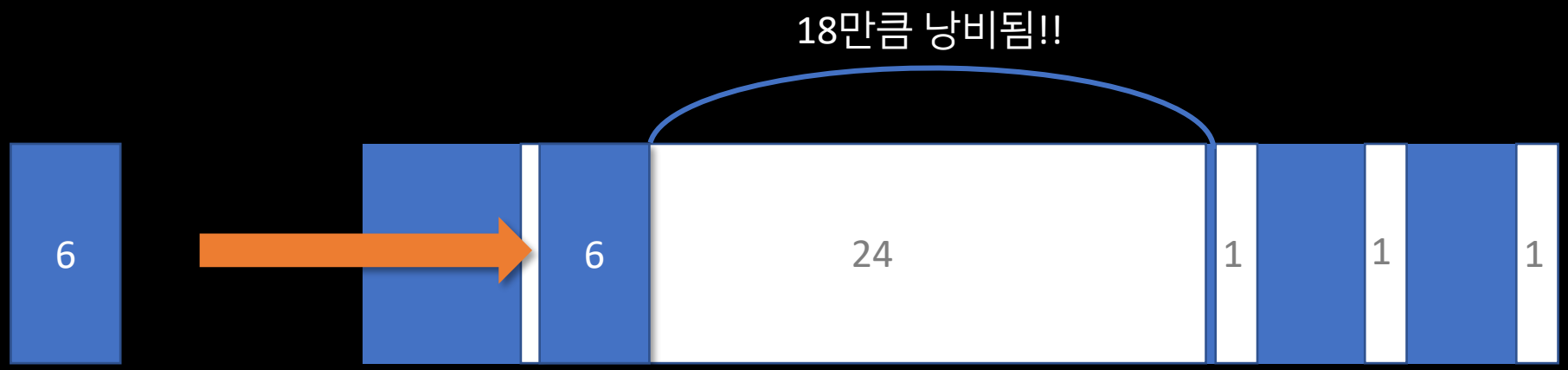
연속할당 방식(가변분할)에서 나타난다.
프로세스가 메모리에 적재되고 제거되면서 메모리의 여유 공간이 여러 조각으로 나뉘어지게 된다.
-> 여유 공간이 충분히 커도 조각나 있으면 연속 메모리 할당이 이루어지지 않아 메모리 낭비가 발생한다.



2. 단편화 Fragmentation

2) 내부 단편화

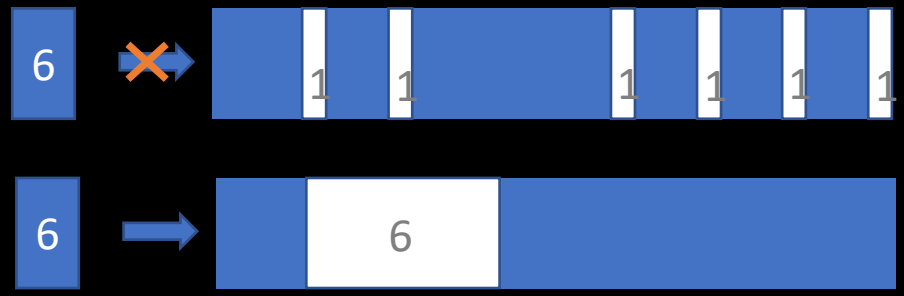
고정분할 방식(파티션)에서 발생한다.
프로세스가 실제로 사용하는 메모리보다 더 큰 메모리가 파티션에 할당되어 낭비되는 현상이다.



2. 단편화 Fragmentation

해결 방법

1. 압축 Compaction
: 외부 단편화를 해소하기 위한 방법으로 hole(자유 공간)을 합친다.
이 과정에서 메모리에 적재된 프로세스는 정지되어야 한다.
ex) GC에서 STW



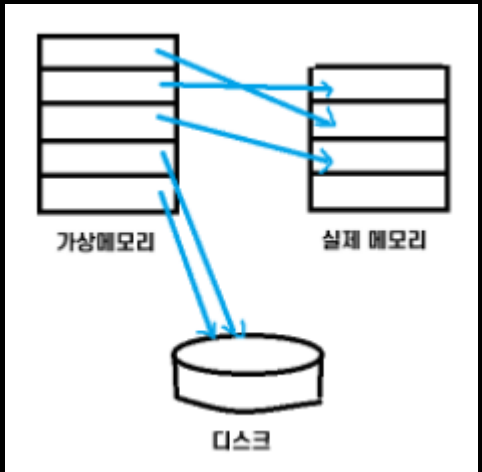
2. 세그멘테이션 Segmentation
: 주소 공간을 의미(기능) 단위의 세그먼트로 나누어 물리적 메모리에 올리는 기법.
세그먼트 크기가 균일하지 않아 외부 단편화가 발생할 수 있다.
ex) 프로세스 메모리 공간

3. 페이징 Paging
: 물리적 메모리를 프레임으로 미리 나누고, 프로세스의 주소 공간을 프레임과 같은 크기의 페이지로 나눈다.
Compaction이 필요하지 않지만 내부 단편화가 발생할 수 있다.

4. 페이지드 세그멘테이션 기법 Paged Segmentation
: 세그멘테이션 + 페이징 기법으로 주소 공간을 의미 단위의 세그먼트로 나누고,
각 세그먼트는 페이지의 집합으로 이루어지게 한다.

3. 가상 메모리

보조 메모리를 캐시로 이용해서 당장 수행할 부분만 메모리에 올려 놓는 방식.
⇒ 지역성의 원칙



가상 주소(virtual address)
: 논리 주소라고 하며, 메모리 관리 장치(MMU)에 의해서 물리 주소로 변환 된다.

물리 주소(physical address)
: 실주소라고도 하며, 실제 메모리 상에서 유효한 주소.

처음에는 가상 주소가 연속적이면 물리 주소도 연속적인 위치에 있게 했지만
페이징 기법을 사용해 좀 더 유연하게 주소 공간을 할당할 수 있게 되었다.

4. 페이지 교체 알고리즘

페이지
: 가상 메모리 블록으로, 요구하는 페이지를 실제 메모리에서 못 찾으면 page fault가 발생.

=> page fault를 최소화하기 위한 여러 가지 페이지 교체 알고리즘이 있음.

1. OPT
: 가장 먼 미래에 참조될 페이지 교체
2. FIFO
: 가장 먼저 올라온 페이지를 교체
3. LRU
: 가장 오래전에 참조가 일어난 페이지 교체
4. LFU
: 참조된 횟수가 가장 적은 페이지부터 교체

References

1~2)
Operating System Concepts 9/E

1~4)
https://snoop-study.tistory.com/92?category=924341

Q&A