**Pintos Project 4: Virtual Memory**

담당 교수 :

이름 / 학번 :

개발 기간 :

1. **개발 목표**

* 해당 프로젝트에서 구현할 내용을 간략히 서술

현재 pintos는 동시에 여러 프로세스를 실행할 수 있다. 하지만 프로세스 초기화 과정에서 program을 memory에 load할 때 별 다른 작업 없이 바로 load하기 때문에 동시에 memory에 load할 수 있는 program의 크기와 개수가 제한되는 상황이다. 이를 해결하기 위해 supplemental page table을 구현하여 프로세스가 필요한 page만을 frame pool로부터 할당 받는 lazy loading을 수행하고, frame pool에 frame이 부족할 때에는 disk swapping을 통해 physical memory의 제약을 극복한다. 이를 위해서는 page fault handling을 새롭게 구현해야 한다. 마지막으로 stack 영역에도 비슷한 정책을 적용하여 더 많은 stack 공간이 필요할 때 허용된 범위 내에서 새로운 공간을 할당하는 stack growth를 구현한다.

1. **개발 범위 및 내용**
   1. **개발 범위**

* 아래 각 항목 개발의 필요성 또는 개발 시 기대되는 결과를 간략히 서술
  1. Page Table & Page Fault Handler  
     기존 page table은 단순히 VPN to PFN mapping 정보만 기록되어 있기 때문에 memory load 여부 등과 같은 부가적인 정보를 담은 supplement page table의 구현이 필요한 상황이다. 또한 이러한 supplement page table을 기반으로 page fault가 발생했을 때 frame pool에서 frame을 할당 받은 후 page table entry의 정보를 기반으로 적절한 작업을 수행함으로써 page fault handling을 구현하여 lazy loading을 정상적으로 수행할 수 있게 된다.
  2. Disk Swap  
     위의 1을 구현하더라도 결국 physical memory size에는 한계가 있기 때문에 frame이 부족한 상황이 생길 수 있다. 이를 해결하기 위해 disk의 일부분을 swap space로 사용한다. 그리고 알고리즘을 이용하여 victim으로 선택된 page의 content를 disk에 저장하거나 다시 필요한 상황이 발생했을 때 disk에서 가져오는 방식으로 disk swapping을 구현한다.
  3. Stack Growth  
     앞서 1, 2를 통해 demand paging을 구현할 수 있게 되는데, stack의 경우도 마찬가지로 이 방식을 따르기 위해 초기에 공간을 크게 할당하지 않는다. 따라서 stack의 확장 또한 구현해야 할 필요성이 있는데, page fault handling 과정에서 stack 확장 여부를 판단하여 추가적인 page를 할당한다.
  4. **개발 내용**
* 아래 항목의 내용만 서술
  1. Page fault가 발생하는 이유와 이를 handling하는 전반적인 과정을 서술  
     Page fault는 유효하지 않은 page에 접근할 때 발생하는데, page가 memory에 없는 경우, 읽기 전용 page에 쓰기를 시도하는 경우, 혹은 정말로 유효한 entry가 존재하지 않는 경우 등의 이유로 발생한다. 이를 handling 하기 위해서는 우선 fault가 발생한 주소가 user program에서 접근할 수 있는 주소인지 확인해야 한다. 만약 접근할 수 없는 주소에 대해 접근을 시도하는 경우 프로세스를 강제 종료한다. 그리고 page entry를 table에서 찾아내고, frame을 할당 받은 후 현재 page의 위치에 따라 file의 내용을 읽어오거나, 0으로 채우거나, disk의 swap space로부터 내용을 읽어온 후 handling을 마무리하여 instruction이 재개될 수 있도록 한다.
  2. Disk swap 발생 시 사용한 page replacement algorithm에 대해 서술  
     LRU의 approximation이라고 할 수 있는 Second Chance Algorithm을 사용하여 victim을 선택한다. 각 page에는 accessed bit이 있고, 이를 확인하거나 수정할 수 있는 함수가 있다. 우선 page들을 쭉 탐색하면서 accessed bit이 0인 page를 발견하면 이를 바로 victim page로 선택한다. 만약 accessed bit이 1이라면 이를 0으로 설정하고 다음 page에 대해 탐색을 진행한다. 이렇게 기회를 한 번 더 부여하는 방식이기 때문에 Second Chance Algorithm이라고 한다.
  3. Stack growth 구현 시 stack 확장 여부를 판단할 수 있는 방법에 대해 서술  
     우선 user program이 접근할 수 있는 주소에서 page fault가 발생하였는데 page table entry를 찾을 수 없다면 stack 확장에 대해 고려해야 한다. 우선 fault가 발생한 주소가 stack의 starting address로부터 stack의 최대 크기 안에 있어야 한다. 일반적으로 stack size의 limit은 8MB이다. 그리고 pintos의 기반인 80x86 system의 PUSHA 연산이 stack pointer로부터 32bytes보다 더 멀리 떨어진 주소에서 발생하면 안된다. 이런 조건을 모두 충족할 경우 stack 확장이 필요한 상황이라고 판단해도 무방하다.

1. **추진 일정 및 개발 방법**
   1. **추진 일정**

* II. A. 개발 범위를 포함하여 구현 내용에 대한 일정 작성
  1. **개발 방법**
* II. B.의 개발 내용을 구현하기 위해 각각에 대해 다음 사항들을 포함하여 설명
  + 수정해야하는 소스코드  
    - vm/frame.c (NEW)  
    - vm/page.c (NEW)  
    - vm/swap.c (NEW)  
    - userprog/process.c
  + 수정하거나 추가해야 하는 자료구조  
    - Supplemental Page Table Entry (struct)  
    - Frame Table Entry (struct)  
    - Swap Table (bitmap)
  + 수정하거나 추가해야 하는 함수  
    - frame\_init : frame table 초기화 함수  
    - frame\_alloc : 새로운 frame을 할당해주는 함수  
    - frame\_free : frame에 대한 자원 해제 함수  
    - frame\_free\_thread : 프로세스 종료 시 프로세스가 사용중이던 frame들에 대해 자원 해제하는 함수  
    - select\_victim\_frame : Second Chance Algorithm을 기반으로 swap out할 frame을 찾는 함수  
    - spt\_init : 각 thread의 supplemental page table에 대한 초기화 함수  
    - spt\_destroy : 각 thread의 supplemental page table에 대한 자원 정리 함수  
    - spt\_find\_entry : 주어진 upage에 해당하는 table entry를 찾아 반환하는 함수  
    - spt\_remove\_entry : 주어진 upage에 해당하는 table entry를 table에서 삭제하는 함수  
    - spt\_create\_filesys : file에 대한 내용을 담는 page에 대한 entry 생성 함수  
    - spt\_create\_zero : zero page에 대한 entry 생성 함수 (stack, …)  
    - swap\_init : swap block을 할당 받고 swap table에 대한 초기화를 수행하는 함수  
    - swap\_out : frame의 내용을 disk에 저장하고 이에 대한 swap index를 반환하는 함수  
    - swap\_in : index에 해당하는 swap page의 내용을 frame에 다시 올리는 함수  
    - swap\_free : swap table에서 index에 해당하는 공간에 대해 사용 가능으로 전환하는 함수

1. **연구 결과**
   1. **Flow Chart**

* II. B. 개발 내용의 각 항목에 대하여 Flow Chart 작성  
  Page fault handling  
  도표, 평면도, 기술 도면, 라인이(가) 표시된 사진

  자동 생성된 설명  
    
  Disk swapping (second chance algorithm)  
  도표, 기술 도면, 평면도, 라인이(가) 표시된 사진

  자동 생성된 설명  
    
  Stack growth  
  도표, 라인, 평면도, 폰트이(가) 표시된 사진

  자동 생성된 설명
  1. **제작 내용**
* II. B. 개발 내용의 각 항목에 대하여 실질적으로 구현한 코드의 관점에서 작성 (구현 내용, 알고리즘 등을 명확히 서술할 것)
  + 구현에 있어 Pintos에 내장된 라이브러리나 자체 제작한 함수를 사용한 경우 이에 대해서도 설명
  1. frame.c  
     텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

     자동 생성된 설명  
     - frame table entry 구조체  
       
     스크린샷, 텍스트, 폰트, 라인이(가) 표시된 사진

     자동 생성된 설명  
     텍스트, 스크린샷, 소프트웨어, 폰트이(가) 표시된 사진

     자동 생성된 설명  
     - frame table 초기화 (hashing 위한 부가적인 함수 정의)  
     - frame table에 대한 lock 초기화  
       
     텍스트, 스크린샷, 소프트웨어, 운영 체제이(가) 표시된 사진

     자동 생성된 설명  
     - frame 할당하는 함수  
     - frame pool에 더 이상 frame이 없다면 victim page 선택하여 disk로 swap out 한 이후 다시 palloc 시도.  
     - frame table entry에 upage, kpage, owner에 대한 정보 기록한 후 frame table에 삽입한 후 kpage 주소 반환.  
       
     텍스트, 스크린샷, 소프트웨어이(가) 표시된 사진

     자동 생성된 설명  
     - frame 할당 해제하는 함수  
     - kpage를 key로 설정하여 해당하는 entry 찾은 후 table에서 제거한 후 palloc 해제, entry 자원 해제.  
       
     텍스트, 스크린샷, 소프트웨어이(가) 표시된 사진

     자동 생성된 설명  
     - thread가 소유한 모든 frame에 대한 자원 해제 함수  
     - owner가 일치한 entry들을 list에 담은 후 한꺼번에 자원 해제  
     - hash의 consistency를 유지하기 위한 방식  
       
     텍스트, 스크린샷, 소프트웨어이(가) 표시된 사진

     자동 생성된 설명  
     - Second Change Algorithm을 기반으로 victim frame 선택하는 함수  
     - pagedir\_is\_accessed, pagedir\_set\_accessed 함수를 이용 (Reference bit 역할)
  2. page.c  
     텍스트, 스크린샷, 폰트, 소프트웨어이(가) 표시된 사진

     자동 생성된 설명  
     - Supplemental page table entry 구조체  
       
     텍스트, 스크린샷, 폰트, 라인이(가) 표시된 사진

     자동 생성된 설명  
     텍스트, 스크린샷, 폰트이(가) 표시된 사진

     자동 생성된 설명  
     - Supplemental page table 초기화 함수  
     - hashing에 필요한 부가적인 함수 정의  
       
     텍스트, 폰트, 스크린샷, 라인이(가) 표시된 사진

     자동 생성된 설명  
     텍스트, 스크린샷, 폰트, 소프트웨어이(가) 표시된 사진

     자동 생성된 설명  
     - Supplemental page table을 destroy 하는 함수  
     - table의 각 entry에 대해서도 자원 해제를 수행하는 함수 선언  
     - swap out 상태인 page인 경우 해당 swap index를 사용 가능 상태로 변경  
       
     텍스트, 스크린샷, 디스플레이, 폰트이(가) 표시된 사진

     자동 생성된 설명  
     - upage에 해당하는 entry를 table에서 찾아 반환하는 함수  
     - 일치하는 entry가 없다면 NULL 반환  
       
     텍스트, 스크린샷, 소프트웨어, 디스플레이이(가) 표시된 사진

     자동 생성된 설명  
     - upage에 해당하는 entry를 table에서 찾아 제거하는 함수  
       
     텍스트, 스크린샷, 소프트웨어이(가) 표시된 사진

     자동 생성된 설명  
     - File에 대한 내용을 담는 page의 entry를 생성하는 함수  
     - 위치를 PAGE\_IN\_FILESYS로 설정  
     - 해당하는 file과 이에 대한 offset, read\_bytes, zero\_bytes 저장  
       
     텍스트, 스크린샷, 디스플레이, 소프트웨어이(가) 표시된 사진

     자동 생성된 설명  
     - 내부를 0으로 채우는 page에 대한 entry를 생성하는 함수  
     - 위치를 PAGE\_ZERO로 설정
  3. swap.c  
     텍스트, 스크린샷, 폰트이(가) 표시된 사진

     자동 생성된 설명  
     - swap block을 할당받고 swap table bitmap을 초기화하는 함수  
       
     텍스트, 스크린샷, 소프트웨어, 디스플레이이(가) 표시된 사진

     자동 생성된 설명  
     - swap table에서 사용 가능한 index를 찾고 이에 해당하는 swap block의 위치에 kpage의 내용 복사  
     - 해당 index를 반환  
       
     텍스트, 스크린샷, 폰트이(가) 표시된 사진

     자동 생성된 설명  
     - index에 해당하는 swap block의 위치에서 내용을 읽어 kpage에 내용 복사  
     - index를 다시 사용 가능 상태로 설정  
       
     텍스트, 스크린샷, 폰트이(가) 표시된 사진

     자동 생성된 설명  
     - swap table의 해당 index에 대해 사용 가능 상태로 설정하는 함수
  4. process.c  
     텍스트, 스크린샷, 소프트웨어, 멀티미디어 소프트웨어이(가) 표시된 사진

     자동 생성된 설명  
     - palloc으로 frame을 바로 할당 받지 않고 supplemental page table entry를 생성하도록 구현 변경  
     - file과 관련된 정보 모두 entry에 저장  
     - 추후 page fault 발생 시 필요한 page를 load 할 수 있게 됨으로써 lazy loading 수행  
       
     텍스트, 스크린샷, 폰트이(가) 표시된 사진

     자동 생성된 설명  
     - 마찬가지로 바로 frame을 할당 받는 것이 아니라 supplemental page table entry를 생성하도록 구현 변경  
     - zero page
  5. process.c (page fault handler)  
     텍스트, 스크린샷, 소프트웨어, 멀티미디어 소프트웨어이(가) 표시된 사진

     자동 생성된 설명  
     - fault가 발생한 page의 주소를 통해 supplemental page table entry를 검색  
     - 만약 entry가 없다면 stack growth가 필요한 상황인지 판단하여 맞다는 이를 수행  
       
     텍스트, 스크린샷이(가) 표시된 사진

     자동 생성된 설명  
     - entry를 찾았다면 해당 page의 위치에 따라 적절한 동작 수행  
     - PAGE\_IN\_FILESYS인 경우 entry에 기록된 file 정보를 기반으로 file read 수행하여 새로 할당 받은 frame에 저장  
     - PAGE\_ZERO인 경우 새로 할당 받은 frame을 0으로 채움  
     - PAGE\_IN\_SWAP인 경우 index에 해당하는 swap page의 내용을 새로 할당 받은 frame에 저장한 후 해당 index를 사용 가능으로 전환  
     - install\_page를 통해 VA와 PA mapping한 후 spte의 location을 PAGE\_IN\_FRAME으로 변경
* 개발 중 발생한 문제나 이슈가 있으면 이를 간략히 설명하고 해결한 방식에 대해 설명  
  Parallel 혹은 synchronization에 대한 test case에 대해서 동시성 오류가 지속적으로 발생하였는데 모든 file operation 코드 부분에 대해 lock을 설정하였다고 생각하여 원인을 찾지 못함. 그러다가 load\_segment 함수의 file operation 부분에서 lock을 설정하지 않은 것을 확인하고 이를 수정하였더니 정상적으로 통과하였음.  
    
  하지만 cspro 환경에서 make check를 수행한 결과 local 환경 (WSL)에서는 정상적으로 통과했던 테스트 케이스 2개가 실패하는 현상이 발생하였음. Lock holder에 관련한 문제로 커널 패닉이 발생하였는데 계속된 디버깅에도 원인을 찾을 수 없어 결국 해결하지 못함.
  1. **시험 및 평가 내용**
* (채점 대상 테스트 케이스에 해당하는) make check 수행 결과를 캡처하여 첨부  
  텍스트, 스크린샷, 폰트, 흑백이(가) 표시된 사진

  자동 생성된 설명