Virtual Memory Management 기법 결과 보고서

이윤영

글로벌경영학과_2019315330

운영체제_SWE3004_01(엄영익)

2022 년 11 월 23 일

과제 목표:

Demand paging system 을 위한 page replacement 기법 구현

세부 내용:

- 주어진 page reference string 을 입력 받아아래의 각 replacement 기법으로 처리했을 경우의 memory residence set 변화 과정 및 page fault 발생 과정 추적/출력
  - ✓ Min
  - ✓ LRU
  - ✓ LFU
  - ✓ WS Memory Management

- Page frame 할당량및 window size 등은입력으로 결정 Deadlock 상태 여부
- 초기 할당된 page frame 들은 모두비어 있는것으로 가정
- 각 기법의 실행 결과에 대한 출력 방법은 각자 design 하여 진행
- 기타 필요한 가정은 각자 설정하고 보고서에 명시

- 입력 포맷 및 입력 예 (입력파일명은 "input.txt")기타 고려사항이나 가정이 필요할 경우, 스스로 판단하여 rule 을 정하고, 이를 결과 보고서에 명시해야 함
  - ✓ N 은 process 가 갖는 page 개수 (최대 100)
  - ✓ M 은 할당 page frame 개수 (최대 20, WS 기법에서는 비사용)
  - ✓ W 는 window size (최대 100, WS 기법에서만 사용)
  - ✓ K 는 page reference string 길이 (최대 1,000)
  - ✓ "r1 r2 r3 r4 r5 … rK"는 page reference string
    - Page 번호(ri)는 0 번부터 시작

설계/구현:

사용한 언어/시스템: Python/Windows 10

(현재 비전공자 학생으로서 유일하게 사용 가능한 언어는 파이썬입니다.)

1. 텍스트 파일 데이터를 리스트에 저장:

   - ✓ 텍스트 파일에 입력된 Page 개수, Page Frame 개수, Window size, Reference string 길이, 그리고 Page Reference String 을 각각 리스트에 담는다

2. MIN 알고리즘 구현:

   - ✓ While 반복문을 사용하여 reference string 리스트가 비어 있게 될 때까지 reference string 을 처음부터 하나씩 제거한다.

   - ✓ 메모리는 비어 있는 상태에서 시작한다.

   - ✓ if 문을 사용하여 해당 페이지가 메모리에 있다면 넘어가고 메모리에 없다면 fault 를 발생시킨다.

   - ✓ Fault 가 발생했을 때 우선 메모리가 비어 있는 상태라면 해당 페이지는 메모리에 곧바로 들어간다.

   - ✓ 메모리에 있는 각 페이지들의 forward distance 를 구해서 forward distance 가 가장 높은 페이지를 victim 으로 선정하고 들어오는 페이지와 교체한다.

   (더 이상 referenced 되지 않아 infinite forward distance 를 갖는 페이지는 reference string 마지막에 등장하는 page 와 동일한 count 수를 갖는 오류가 발생하여 이를 해결하기 위해 if 문을 넣어 메모리에 page 가 infinite forward distance 를 갖는 경우 곧바로 victim 으로 선정하였다. 해당 오류는 LRU 기법이나 LFU 기법에서는 발생하지 않는데, 그 이유는 LRU 기법과 LFU 기법은 referenced history 를 확인하여 backward distance 를 구하기 때문에 메모리에 있는 페이지는 무조건 적어도 한번 이상은 referenced 되었기 때문이다.)

   - ✓ 두개 이상의 페이지가 infinite forward distance 를 갖거나 동일한 forward distance 를 갖는 경우, tie-breaking rule 을 설정하여 첫 번째 페이지, 즉 가장 늦게 참조된 페이지가 victim 으로 선정된다.

   - ✓ Page Fault 가 발생할 때마다 Page Fault 리스트에 해당 페이지를 추가하고, n 값이 1 씩 증가하게 함으로써 page fault 가 발생했을 때의 시간(Time)을 기록한다.

3. LRU 알고리즘 구현:

- ✓ MIN 알고리즘과 비슷하게 while 반복문을 사용하여 reference string 리스트가 비어 있게 될 때까지 reference string 을 처음부터 하나씩 제거한다.
- ✓ 메모리는 비어 있는 상태에서 시작하고, if 문을 사용하여 해당 페이지가 메모리에 있다면 넘어가고 메모리에 없다면 fault 를 발생시킨다.
- ✓ 페이지가 처리될 때마다 reference history 리스트에 추가하고 해당 리스트를 사용하여 fault 발생시 메모리에 있는 page 들의 backward distance 를 구한다.
- ✓ Backward distance 는 reference history 를 reverse 시키고 차례대로 각 page 가 나올 때까지 count 하여 구한다.
- ✓ Backward distance 가 가장 높은 page 를 victim 으로 선정하고 들어오는 페이지와 메모리에서 교체된다.
- ✓ Page Fault 가 발생할 때마다 Page Fault 리스트에 해당 페이지를 추가하고, n 값이 1 씩 증가하게 함으로써 page fault 가 발생했을 때의 시간(Time)을 기록한다.

4. LFU 알고리즘 구현:

- ✓ While 반복문을 사용하여 reference string 리스트가 비어 있게 될 때까지 reference string 을 처음부터 하나씩 제거한다.
- ✓ 메모리는 비어 있는 상태에서 시작하고, if 문을 사용하여 해당 페이지가 메모리에 있다면 넘어가고 메모리에 없다면 fault 를 발생시킨다.
- ✓ reference string 리스트로부터 page number 리스트를 만들고 각 page number 리스트가 reference string 에서 몇 번 참조되는지 count 하여 reference count 리스트를 생성한다.
- ✓ Fault 가 발생할 때마다 메모리에 page 들의 reference count 를 비교하여 가장 낮은 reference count 를 갖는 page 를 victim 으로 선정하고 메모리에 교체한다.
- ✓ 만약, reference count 가 가장 낮지만 동일한 page 가 두개 이상일때는 tie-breaking rule 로 LRU 기법을 사용하여 victim 을 선정하고 들어오는 page 와 교체된다.

- ✓ Page Fault 가 발생할 때마다 Page Fault 리스트에 해당 페이지를 추가하고, n 값이 1 씩 증가하게 함으로써 page fault 가 발생했을 때의 시간(Time)을 기록한다.

5. Working-Set 알고리즘 구현:

- ✓ While 반복문을 사용하여 reference string 리스트가 비어 있게 될 때까지 reference string 을 처음부터 하나씩 제거한다.

- ✓ 메모리는 비어 있는 상태에서 시작하고, if 문을 사용하여 해당 페이지가 메모리에 있다면 넘어가고 메모리에 없다면 fault 를 발생시킨다.

- ✓ Reference history 리스트를 생성하여 페이지가 하나씩 제거될 때마다 해당 리스트에 추가한다.

- ✓ window size, 즉 delta 값은 현재 참조되는 페이지를 포함하여 전에 참조되었던 페이지들을 확인하기 때문에 reference history 리스트의 길이가 window size+ 1 의 크기가 되었을 때 reference history 리스트에서 첫번째 요소를 제거한다.

- ✓ Page fault 발생 시 해당 페이지를 메모리에 추가한다.

- ✓ if 문을 통해 만약에 메모리 안에 존재하는 페이지가 reference history 리스트에 안에 존재하지 않으면 해당 페이지를 제거한다.

- ✓ Page Fault 가 발생할 때마다 Page Fault 리스트에 해당 페이지를 추가하고, n 값이 1 씩 증가하게 함으로써 page fault 가 발생했을 때의 시간(Time)을 기록한다.

<u>설정한 가정:</u>

- ✓ MIN 알고리즘에서 두개 이상의 페이지가 infinite forward distance 를 갖거나 동일한 forward distance 를 갖는 경우, tie-breaking rule 을 설정하여 첫 번째 페이지, 즉 가장 늦게 참조된 페이지가 victim 으로 선정된다.

- ✓ 입력 포맷:

```
6 3 3 15
0 1 2 3 2 3 4 5 4 1 3 4 3 4 5
```

- 위 세부 내용에서 보여준 입력 포맷 예시와 같이 텍스트 파일 데이터는 두줄로 이루어져 있어야 하며, 첫 번째 줄은 차례대로 Page 개수, Page Frame 개수, Window size, Reference string 길이를 담고 두 번째 줄에는 Page Reference String 가 입력된다.

- 각 숫자 사이에는 띄어쓰기가 필요하다.

실행결과:

- 각 알고리즘 기법의 결과에는 Page Fault 총 횟수와 메모리 상태 변화과정이 출력된다.
- 메모리 상태 변화과정은 Page Fault 가 발생한 시간 (Time)을 보여준다.
- 단, Time 은 0 이 아닌 1 부터 시작된다. 즉, Reference string 의 첫번째 페이지는 Time 1 에 속한다.)

예시 1)

입력:

```
6 3 3 15
0 1 2 3 2 3 4 5 4 1 3 4 3 4 5
```

출력:

```
Min 알고리즘 기법 결과------------------------------------------------------------
Page Fault 총 횟수: 7
메모리 상태 변화과정(Time): [1, 2, 3, 4, 7, 8, 11]

LRU 알고리즘 기법 결과------------------------------------------------------------
Page Fault 총 횟수: 9
메모리 상태 변화과정(Time): [1, 2, 3, 4, 7, 8, 10, 11, 15]

LFU 알고리즘 기법 결과------------------------------------------------------------
Page Fault 총 횟수: 8
메모리 상태 변화과정(Time): [1, 2, 3, 4, 7, 8, 10, 15]

Working Set 알고리즘 기법 결과----------------------------------------------------------
Page Fault 총 횟수: 7
메모리 상태 변화과정(Time): [1, 2, 3, 4, 7, 8, 10]
```

예시 2)

입력:

```
5 3 3 10
2 2 3 1 2 4 2 4 0 3
```

출력:

```
Min 알고리즘 기법 결과------------------------------------------------------------
Page Fault 총 횟수: 5
메모리 상태 변화과정(Time): [1, 3, 4, 6, 9]

LRU 알고리즘 기법 결과------------------------------------------------------------
Page Fault 총 횟수: 6
메모리 상태 변화과정(Time): [1, 3, 4, 6, 9, 10]

LFU 알고리즘 기법 결과------------------------------------------------------------
Page Fault 총 횟수: 6
메모리 상태 변화과정(Time): [1, 3, 4, 6, 9, 10]

Working Set 알고리즘 기법 결과----------------------------------------------------------
Page Fault 총 횟수: 6
메모리 상태 변화과정(Time): [1, 3, 4, 6, 9, 10]
```

예시 3)

입력:

```
5 4 3 14
1 2 6 1 4 5 1 2 1 4 5 6 4 5
```

출력:

```
Min 알고리즘 기법 결과------------------------------------------------------------
Page Fault 총 횟수: 6
메모리 상태 변화과정(Time): [1, 2, 3, 5, 6, 12]

LRU 알고리즘 기법 결과------------------------------------------------------------
Page Fault 총 횟수: 7
메모리 상태 변화과정(Time): [1, 2, 3, 5, 6, 8, 12]

LFU 알고리즘 기법 결과------------------------------------------------------------
Page Fault 총 횟수: 7
메모리 상태 변화과정(Time): [1, 2, 3, 5, 6, 8, 12]

Working Set 알고리즘 기법 결과----------------------------------------------------------
Page Fault 총 횟수: 7
메모리 상태 변화과정(Time): [1, 2, 3, 5, 6, 8, 12]
```