

5주차 2차시

운영체제(2)

컴퓨터 시스템과 사용자 간의 중개자_운영 체제의 개념과 동작 원리

- 1 CPU 스케줄링
- 2 기억장치 관리
- 3 정보 관리



운영체제(2)

◆ 학습목표

- 운영체제의 자원 관리 기법을 공부한다.

1. CPU 스케줄링

1) CPU 스케줄링의 목적

- ◆ 스케줄링의 공정성
- ◆ 처리량 최대화
- ◆ 응답 시간 최소화
- ◆ 반환 시간 예측 가능
- ◆ 자원의 균형적인 사용
- ◆ 응답 시간과 자원 활용도의 조화
- ◆ 프로세스 실행의 무한 연기 배제
- ◆ 우선순위에 따른 실행 보장
- ◆ 시스템의 과도한 부하 방지

1. CPU 스케줄링

2) CPU 스케줄링 기법

◆ FCFS

- 준비 큐에 도착한 프로세스 순서대로 CPU를 할당하는 방식
- 비선점 방식

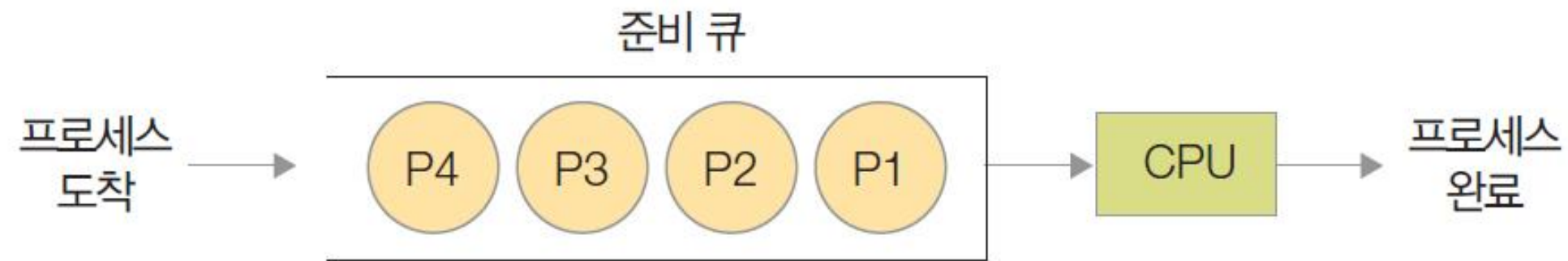


그림 5-19 FCFS 스케줄링

1. CPU 스케줄링

2) CPU 스케줄링 기법

◆ 라운드 로빈

- FCFS 기법과 동일하게 준비 큐에 도착한 프로세스 순서대로 CPU를 할당하지만, 프로세스마다 CPU 사용 시간을 일정하게 할당하여 시간이 지나면 다음 프로세스에게 CPU 사용권을 넘김
- 선점 방식



그림 5-20 라운드 로빈 스케줄링

1. CPU 스케줄링

2) CPU 스케줄링 기법

◆ SJF

- 준비 큐에서 대기하고 있는 프로세스 중 예상 실행 시간이 가장 짧은 것을 우선 처리
- 비선점 방식

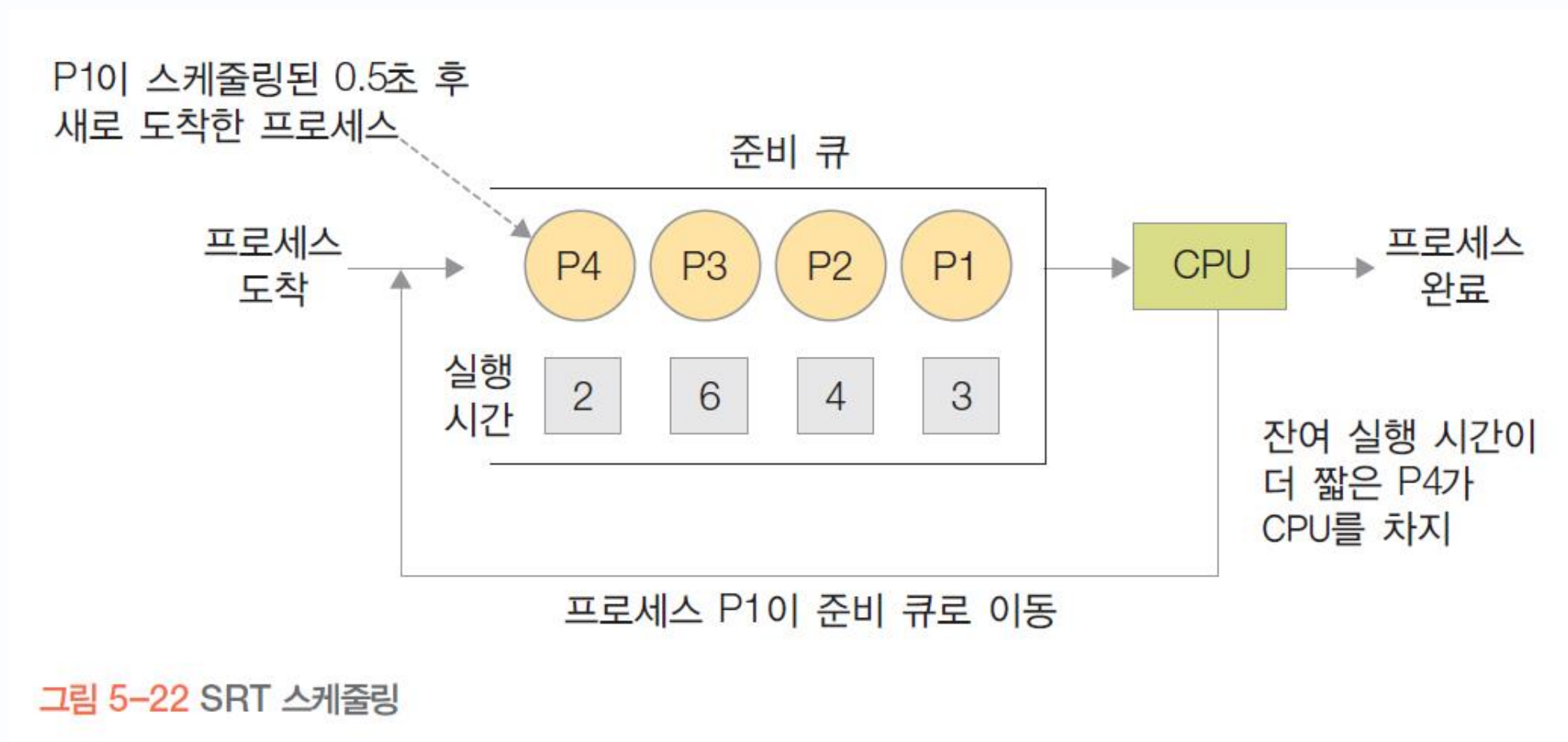


1. CPU 스케줄링

2) CPU 스케줄링 기법

◆ SRT

- 실행 중인 프로세스의 잔여 실행 시간과 준비 큐에서 대기 중인 프로세스의 잔여 실행 시간을 비교하여 가장 짧은 프로세스에게 CPU를 할당
- 선점 방식



1. CPU 스케줄링

2) CPU 스케줄링 기법

◆ HRN

- 실행 시간이 길더라도 대기를 오래 했다면 우선 처리될 수 있도록 스케줄링 할 때 매번 프로세스의 응답률(우선순위)을 계산
- 응답률 공식

$$\text{응답률} = (\text{대기 시간} + \text{실행 시간}) / \text{실행 시간}$$
- 비선점 방식



1. CPU 스케줄링

2) CPU 스케줄링 기법

◆ 우선 순위

- 프로세스마다 우선순위를 부여해 최상위 우선순위 프로세스부터 처리하는 방식
- 비선점 방식



그림 5-24 우선순위 스케줄링

1. CPU 스케줄링

3) CPU 스케줄링 기법의 성능 기준

- ◆ CPU 활용률
- ◆ 처리율
- ◆ 반환 시간
- ◆ 대기 시간
- ◆ 응답 시간

2. 기억장치 관리

1) 주기억장치 관리 기법

◆ 단일 프로그래밍 기법

- 운영체제와 하나의 사용자 프로그램만 주기억장치에 적재
- 주기억장치의 빈 공간이 있어도 다른 사용자 프로그램을 실행할 수 없음
- 자원 낭비가 심함
- 주기억장치의 용량보다 작은 프로그램만 실행



그림 5-27 단일 프로그래밍 기법

2. 기억장치 관리

1) 주기억장치 관리 기법

◆ 고정 분할 다중 프로그래밍 기법

- 주기억장치를 다수의 고정된 크기로 나눠서 실행 중인 여러 프로세스에게 할당
- 분할은 하나의 작업이 적재될 수 있는 일정한 크기의 기억 공간

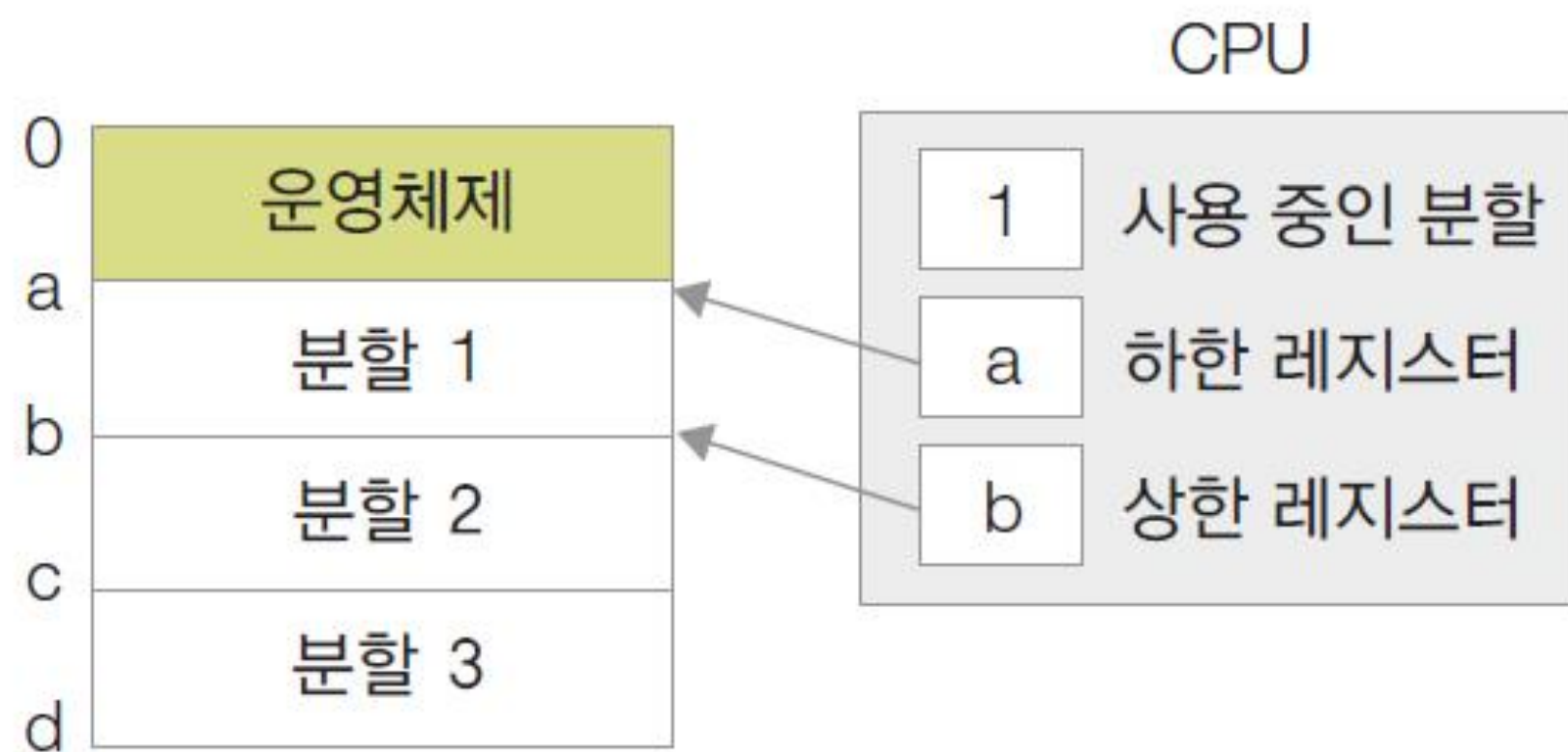


그림 5-28 고정 분할 기억장치의 보호

2. 기억장치 관리

1) 주기억장치 관리 기법

◆ 고정 분할 다중 프로그래밍 기법

- 작업과 분할의 크기가 일치하지 않아 사용되지 않는 빈 공간이 생기거나, 분할이 너무 작아서 작업을 적재하지 못하는 단편화 현상이 발생
 - 내부 단편화 : 분할에 작업을 적재한 후 빈 공간이 남는 현상
 - 외부 단편화 : 적재할 작업보다 분할의 크기가 작아서 분할이 빈 공간으로 남는 현상



그림 5-29 고정 분할 기억장치의 단편화 현상

2. 기억장치 관리

1) 주기억장치 관리 기법

◆ 가변 분할 다중 프로그래밍 기법

- 고정 분할 다중 프로그래밍 기법의 단점을 보완한 기법
- 고정된 분할의 경계를 없애고 각 작업에게 필요한 만큼의 기억 공간을 할당
- 작업이 완료되면 사용되지 않는 기억 공간을 회수하여 관리



그림 5-30 가변 분할 기억장치의 기억 공간 할당

2. 기억장치 관리

1) 주기억장치 관리 기법

◆ 가변 분할 다중 프로그래밍 기법

- 새로운 작업의 적재와 완료된 작업의 기억 공간 회수가 반복됨에 따라 단편화 현상이 발생



그림 5-31 가변 분할 기억장치의 단편화 현상

2. 기억장치 관리

1) 주기억장치 관리 기법

◆ 가변 분할 다중 프로그래밍 기법

- 통합 : 인접한 빈 공간을 합병하여 하나의 큰 공백을 형성하는 기법
- 집약 : 주기억장치의 한쪽 끝에 하나의 큰 공백을 형성하는 기법



그림 5-32 가변 분할 기억장치의 공백 통합



그림 5-33 가변 분할 기억장치의 공백 집약

2. 기억장치 관리

2) 주기억장치 관리 전략

◆ 반입 전략

- 다음에 실행할 프로그램이나 참조할 데이터를 보조기억장치에서 주기억장치로 언제 가져올지 결정하는 전략
- 요구 반입 전략
- 예상 반입 전략

2. 기억장치 관리

2) 주기억장치 관리 전략

◆ 배치 전략

- 새로 가져온 프로그램이나 데이터를 주기억장치의 어디에 배치할지 결정하는 전략
- 최초 적합

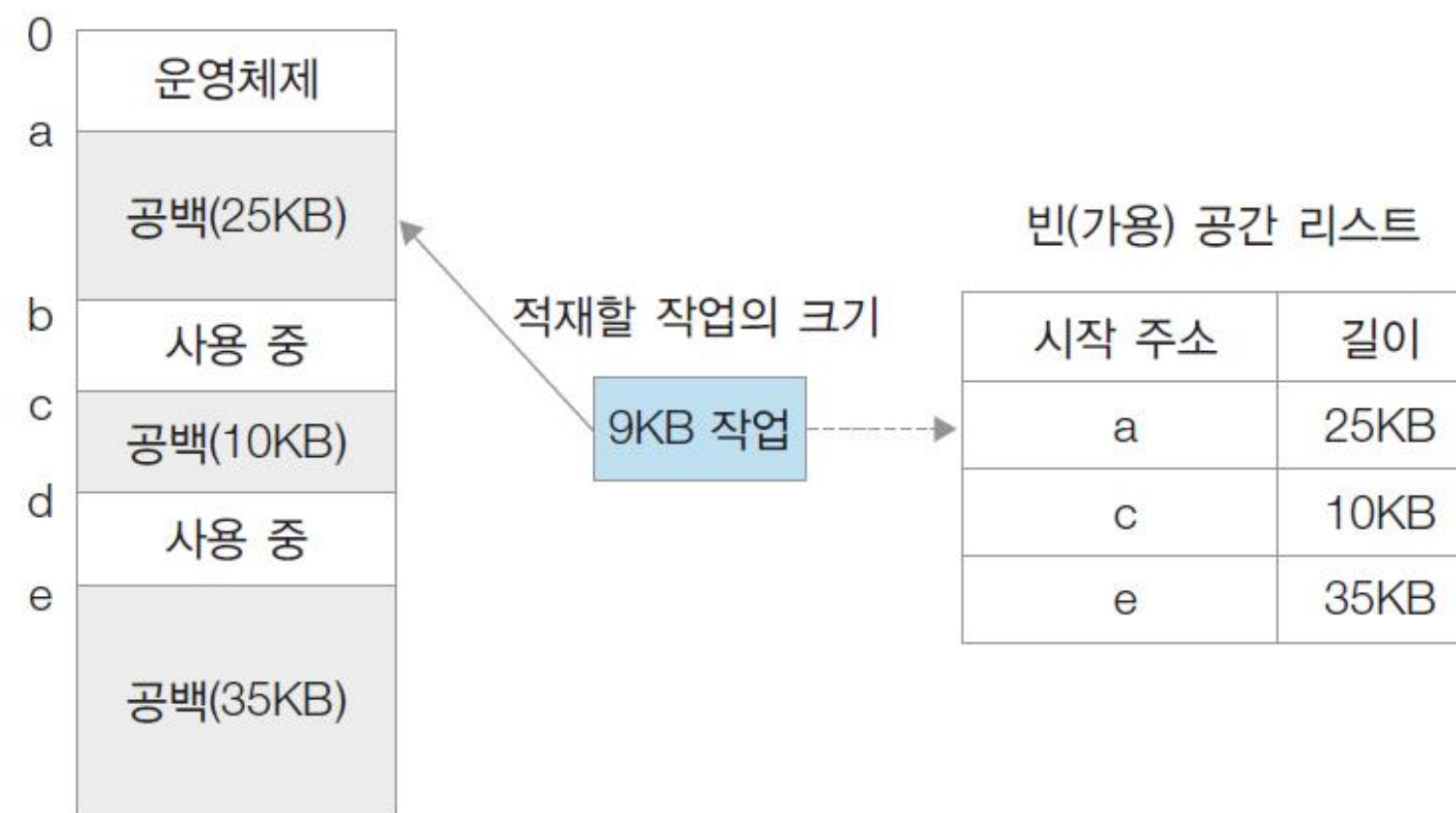


그림 5-34 최초 적합

2. 기억장치 관리

2) 주기억장치 관리 전략

◆ 배치 전략

- 새로 가져온 프로그램이나 데이터를 주기억장치의 어디에 배치할지 결정하는 전략
- 최적 적합

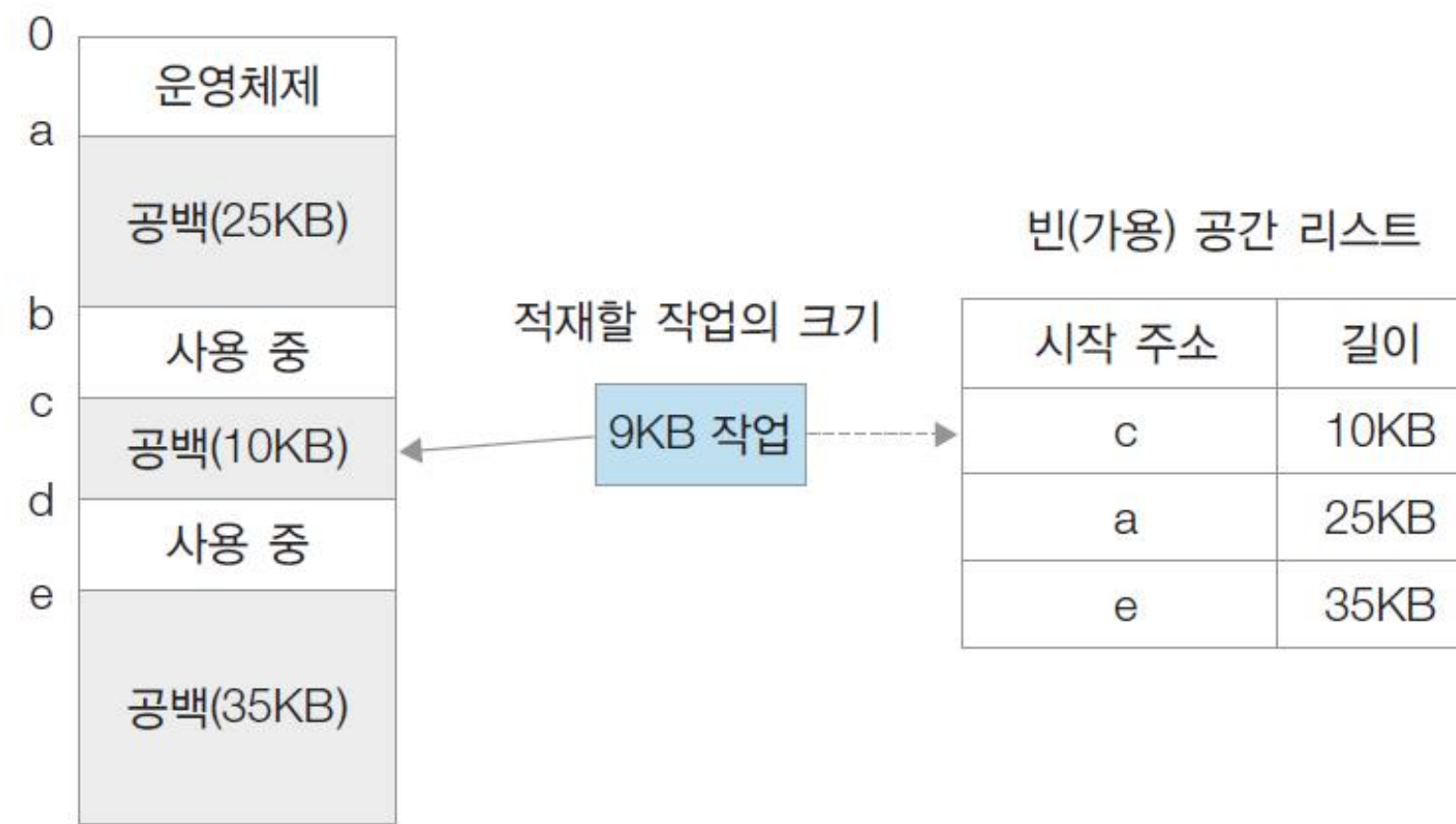


그림 5-35 최적 적합

2. 기억장치 관리

2) 주기억장치 관리 전략

◆ 배치 전략

- 새로 가져온 프로그램이나 데이터를 주기억장치의 어디에 배치할지 결정하는 전략
- 최악 적합

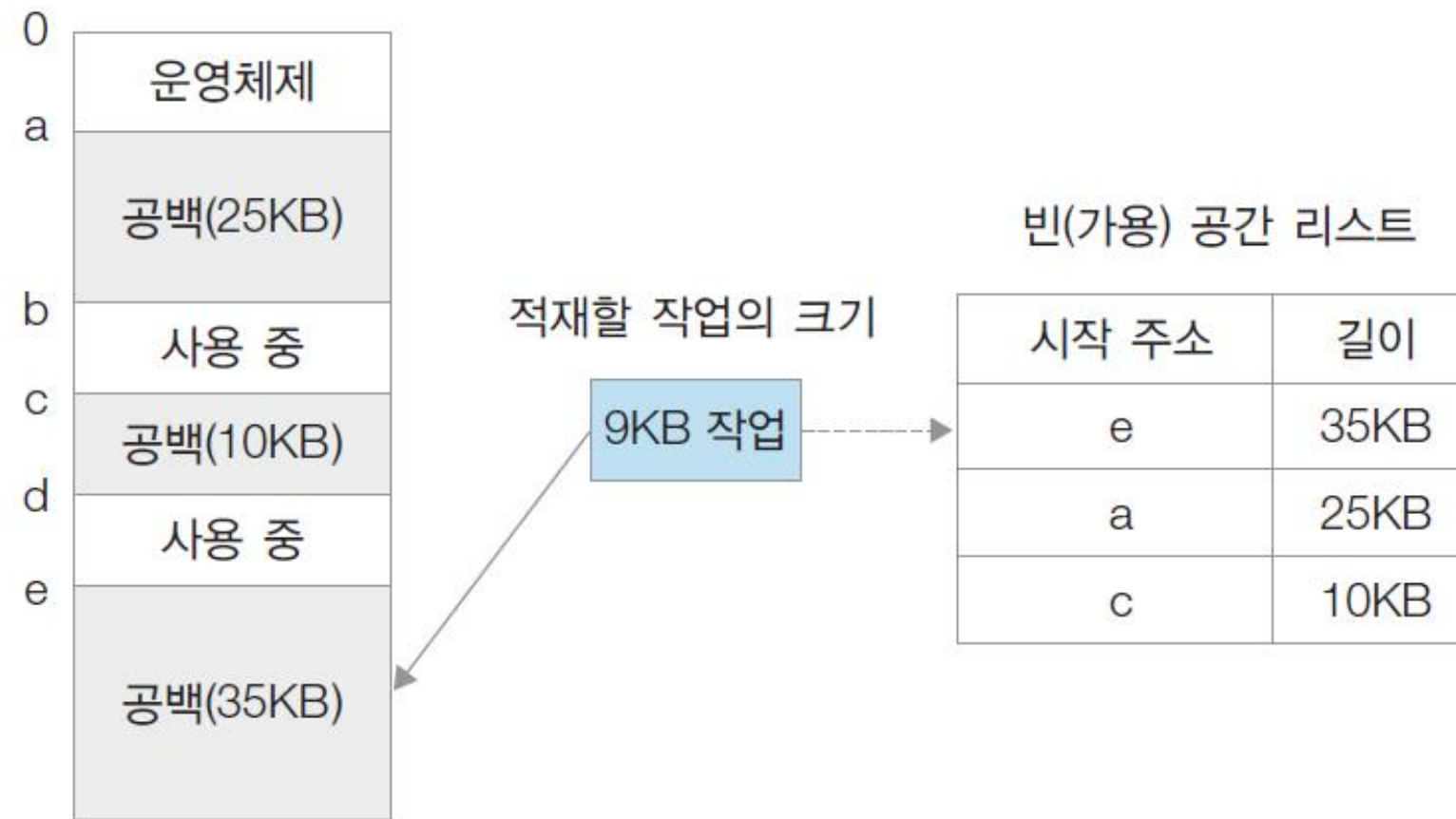


그림 5-36 최악 적합

2. 기억장치 관리

2) 주기억장치 관리 전략

◆ 교체 전략

- 주기억장치에 새로 반입할 프로그램이나 데이터를 배치할 빈 공간이 없을 때, 배치 공간을 마련하기 위해 주기억장치에서 어떤 프로그램이나 데이터를 제거할지 결정하는 전략
- 최적화 원리, 무작위 교체, 선입 선출 교체, LRU 교체, NUR 교체, LFU 교체, 2차 기회 교체 등

2. 기억장치 관리

3) 가상기억장치 구현 기법

◆ 가상기억장치

- 현재 실행 중인 프로그램의 일부는 주기억장치에 적재하고, 나머지는 보조기억 장치에 유지
- 프로세스가 참조하는 가상 주소는 주소 사상 함수에 의해 실행 중에 주기억장치에서 사용할 수 있는 실제 주소로 변환하는 동적 주소 변환 과정을 거쳐야 함

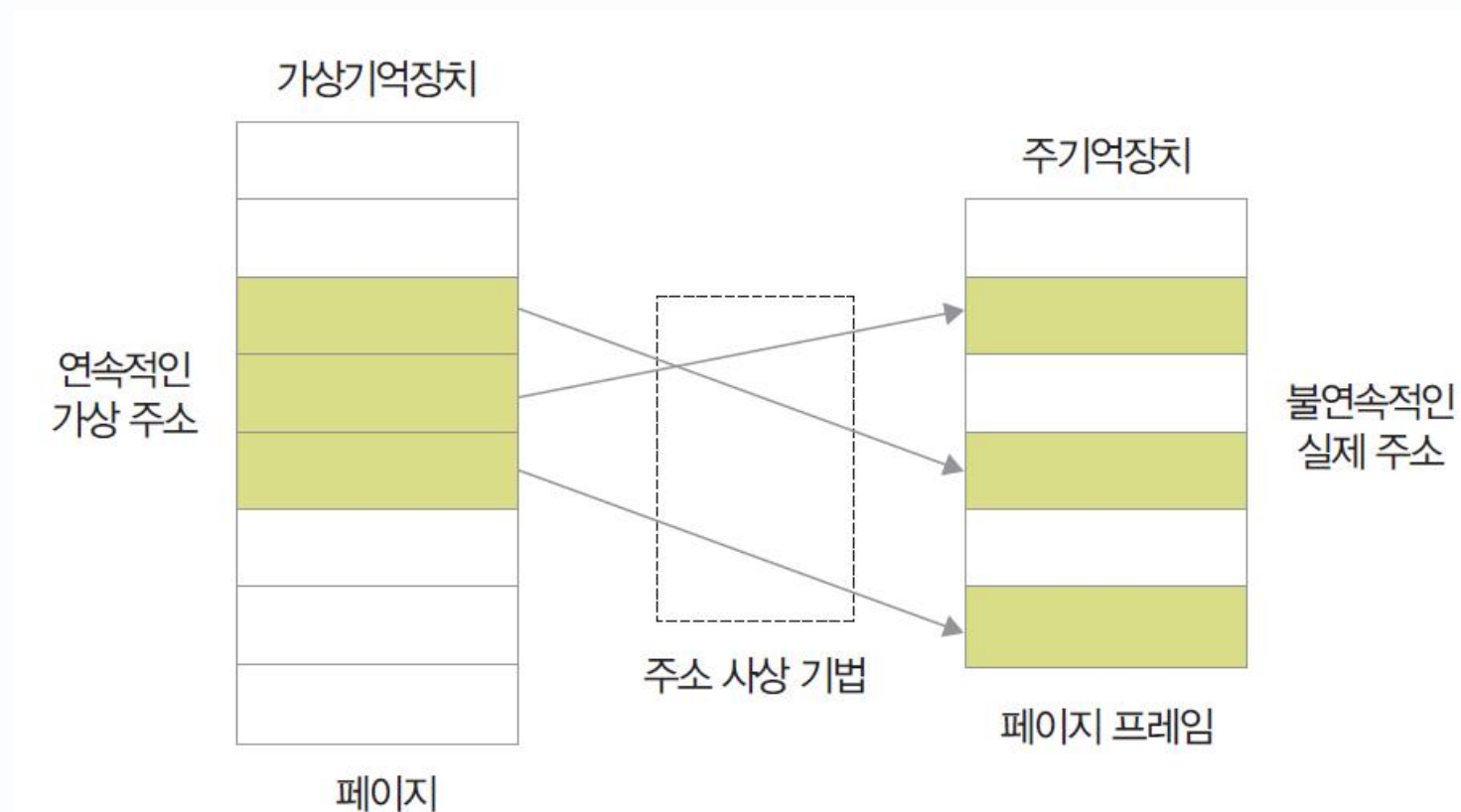


그림 5-37 가상기억장치에서 주기억장치로의 동적 주소 변환

2. 기억장치 관리

3) 가상기억장치 구현 기법

◆ 페이징 기법

- 가상기억장치를 일정한 크기의 페이지로 나눠서 관리
- 주기억장치는 페이지와 크기가 동일한 페이지 프레임으로 분할하여 사용
- 가상 주소는 $V=(p, d)$
- 프로세스마다 별도의 페이지 테이블을 가짐

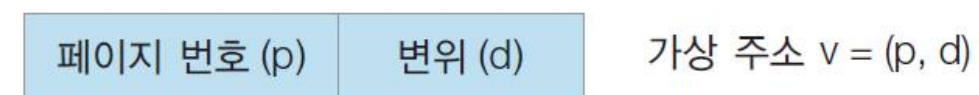


그림 5-38 페이징 기법의 가상 주소 형식

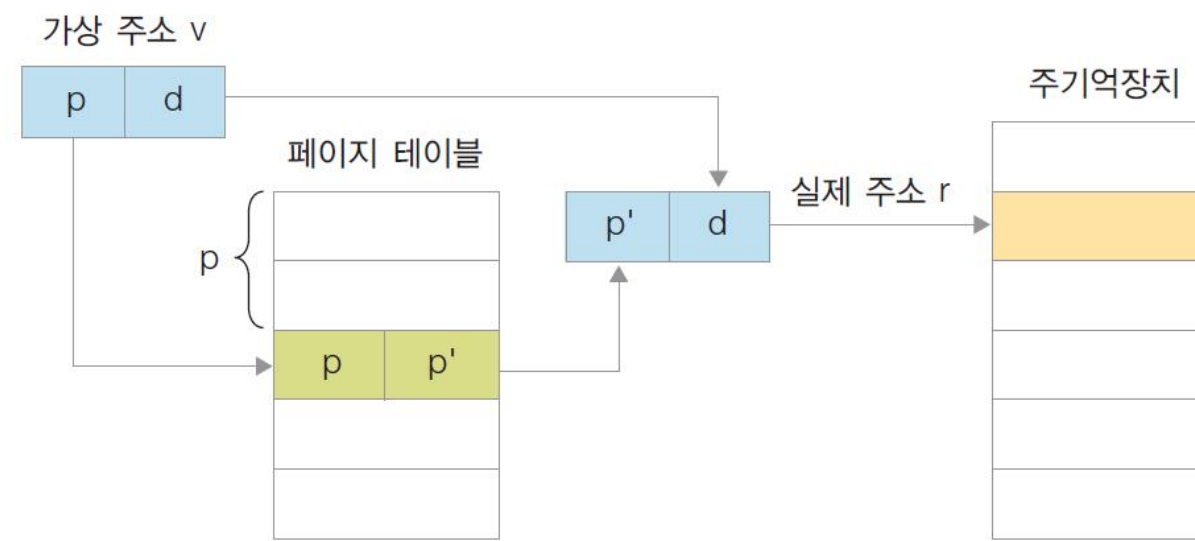


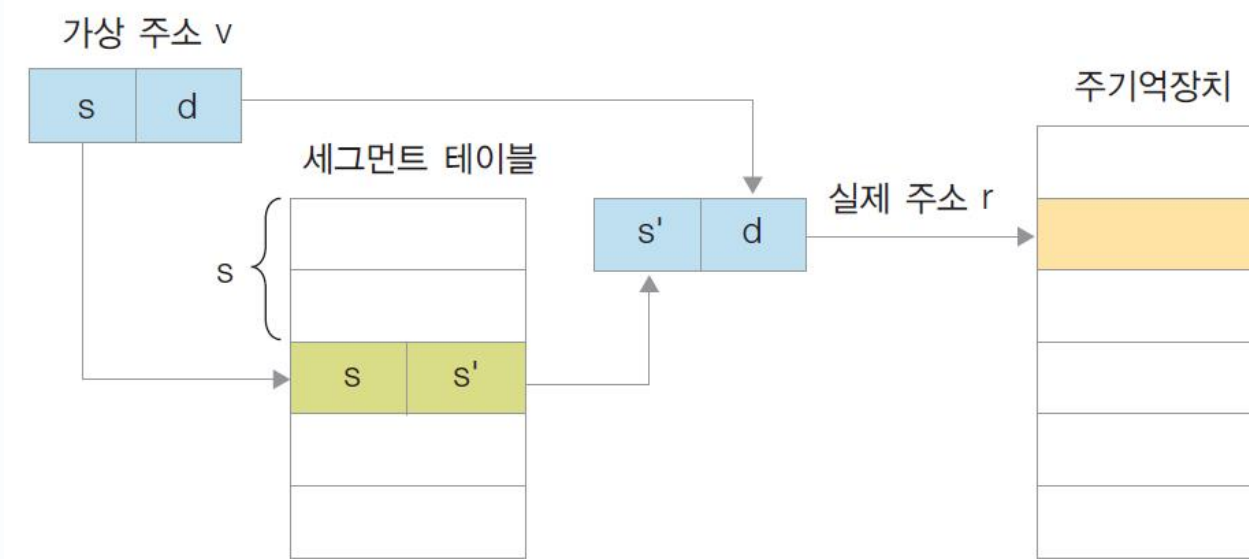
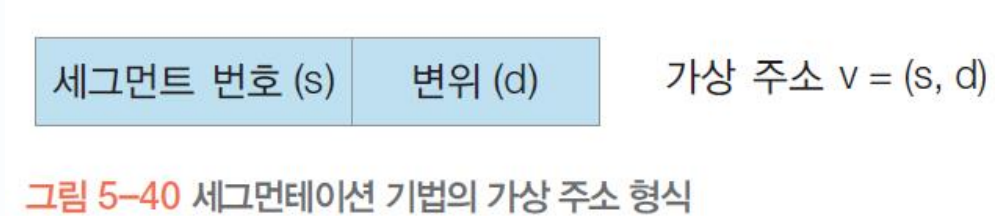
그림 5-39 페이징 기법의 주소 변환 과정

2. 기억장치 관리

3) 가상기억장치 구현 기법

◆ 세그멘테이션 기법

- 프로그램이나 데이터를 용도에 따라 가변적인 크기로 분할하여 관리
- 가변적인 크기로 분할된 프로그램이나 데이터 블록을 세그먼트
- 가상 주소는 $V=(s, d)$
- 프로세스마다 별도의 세그먼트 테이블을 가짐



2. 기억장치 관리

3) 가상기억장치 구현 기법

◆ 페이징/세그멘테이션 혼합 기법

- 모든 세그먼트를 페이지 단위로 다시 분할해 외부 단편화 현상을 해결
- 세그먼트의 크기는 페이지의 정수 배
- 가상 주소는 $V=(s, p, d)$

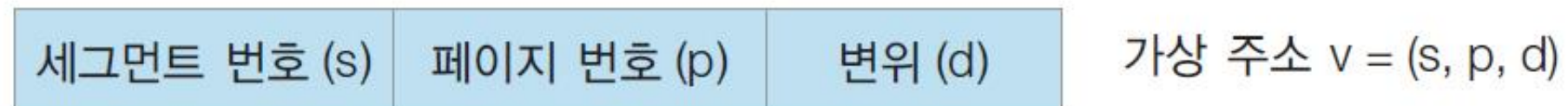


그림 5-42 페이징/세그멘테이션 혼합 기법의 가상 주소 형식

2. 기억장치 관리

3) 가상기억장치 구현 기법

◆ 페이징/세그멘테이션 혼합 기법

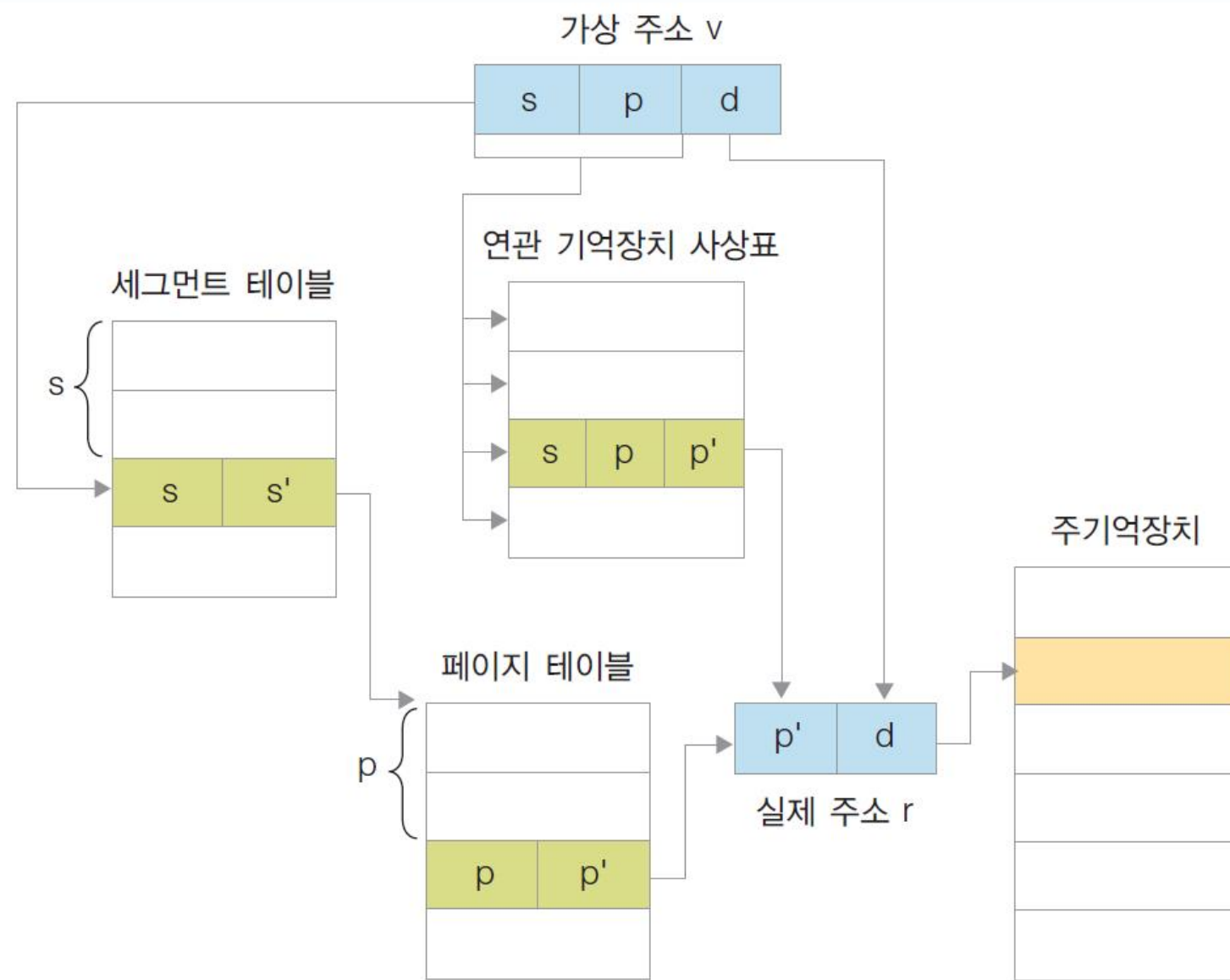


그림 5-43 페이징/세그멘테이션 혼합 기법의 주소 변환 과정

3. 정보 관리

1) 파일 시스템

◆ 파일 시스템의 기능

- 사용자에게 편리한 사용자 인터페이스를 제공
- 사용자가 파일을 생성, 수정, 삭제할 수 있는 기능을 제공
- 적절한 제어 방법을 통해 다른 사람의 파일을 공유할 수 있는 기능을 제공
- 불의의 사고에 대비하기 위한 파일 백업 및 복구 기능을 제공
- 정보를 암호화하고 해독할 수 있는 능력을 제공

3. 정보 관리

2) 파일 구조

◆ 순차 파일

- 레코드를 순차적으로 저장하고 검색
- 장점 : 레코드 사이에 빈 공간이 없어 기억장치를 효율적으로 이용함
- 단점 : 레코드의 삽입과 삭제가 어렵고 파일 검색에 많은 시간이 소요됨

◆ 색인 순차 파일

- 키 값에 따라 정렬된 레코드를 순차적으로 접근하는 순차 파일 구조와 키 값에 따라 직접적으로 접근하는 직접 파일 구조가 모두 사용
- 데이터 파일과 색인 파일로 구성
- 장점 : 비교적 레코드에 빠르게 접근할 수 있으며 레코드의 삽입과 수정이 용이
- 단점 : 처리 속도가 늦고, 색인을 저장하는 공간이 별도로 필요함

3. 정보 관리

2) 파일 구조

◆ 직접 파일

- 특정 레코드에 직접 접근할 수 있는 파일 형태
- 레코드가 저장된 보조기억장치의 물리 주소를 키 값으로 계산하여 접근
- 장점 : 처리 속도가 빠르고, 다른 레코드에 영향을 주지 않으면서 특정 레코드를 삽입, 삭제, 검색, 수정할 수 있음
- 단점 : 키 값에 의한 순차 검색이 어려움

3. 정보 관리

2) 파일 구조

◆ 분할 파일

- 다수의 순차적인 서브 파일로 구성
- 프로그램 라이브러리나 매크로 라이브러리를 저장할 때 유용
- 순차적인 서브 파일의 시작 주소는 디렉터리 레코드에 저장

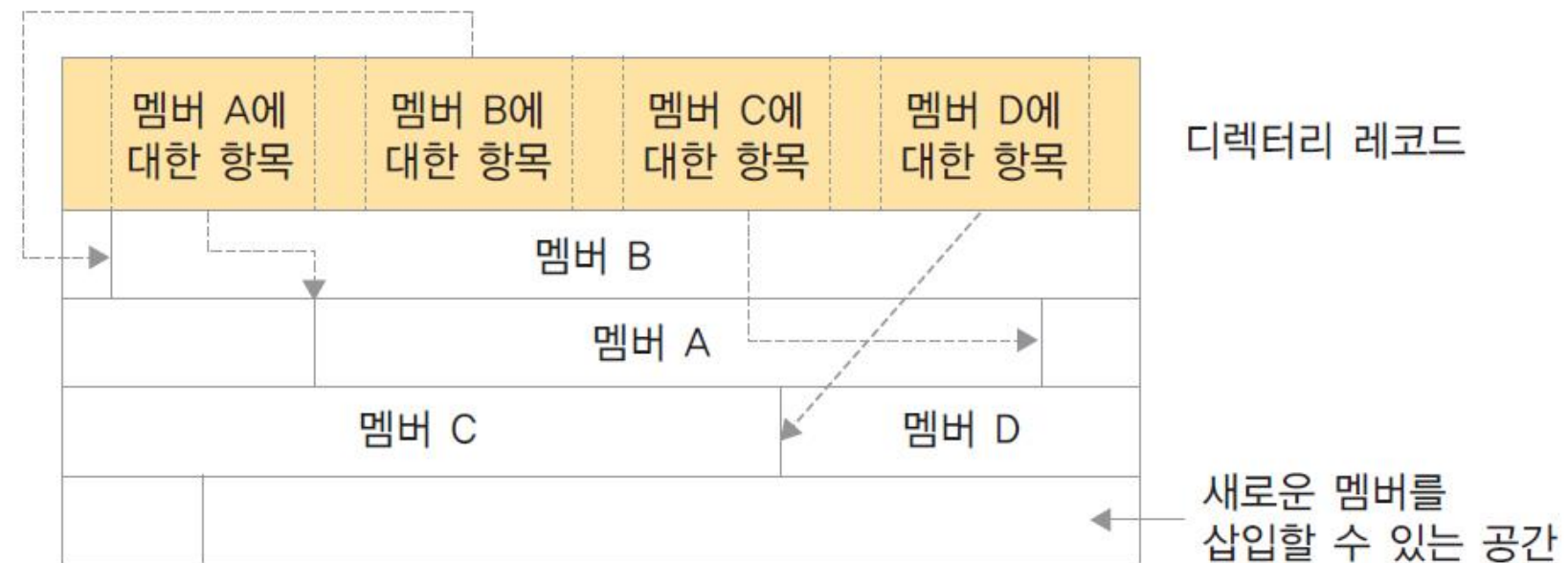


그림 5-44 분할 파일

3. 정보 관리

3) 파일의 공간 할당 기법

◆ 연속 파일

- 파일을 디스크 내의 연속적인 빈 공간에 저장
- 논리적으로 연속된 레코드들이 물리적으로도 인접하게 저장되어 접근 시간이 감소
- 디렉터리에는 파일을 구성하는 첫 번째 블록의 시작 주소와 길이만 유지하면 되므로 구현이 쉬움
- 주기적인 집약이 필요

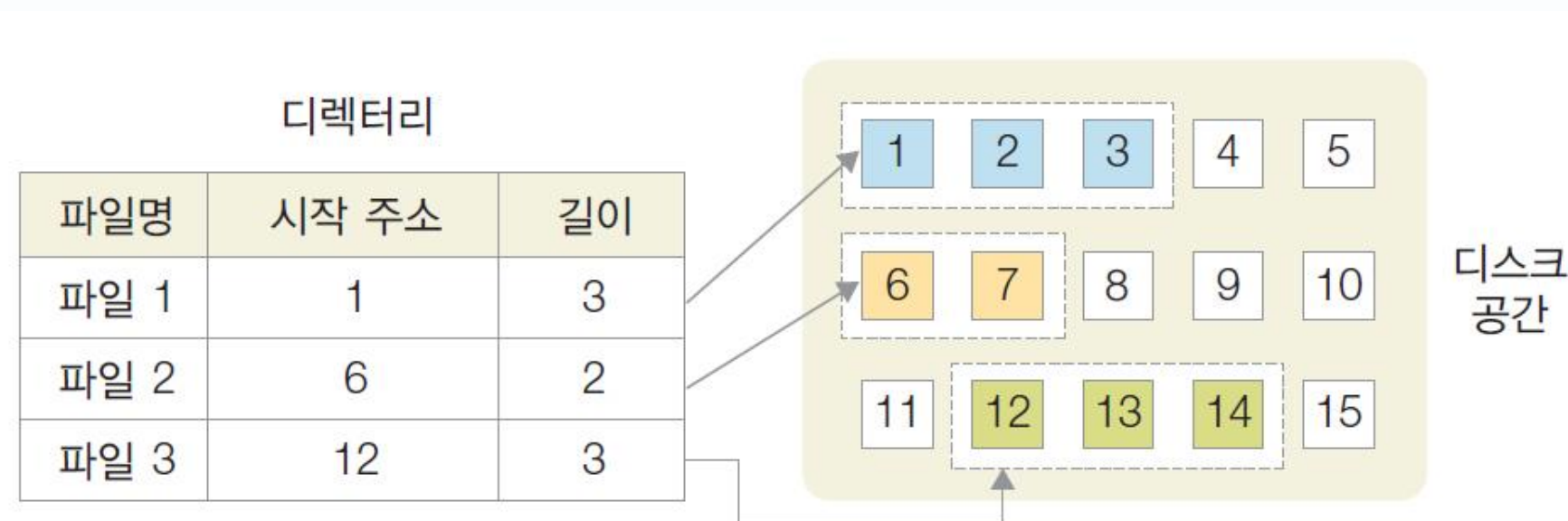


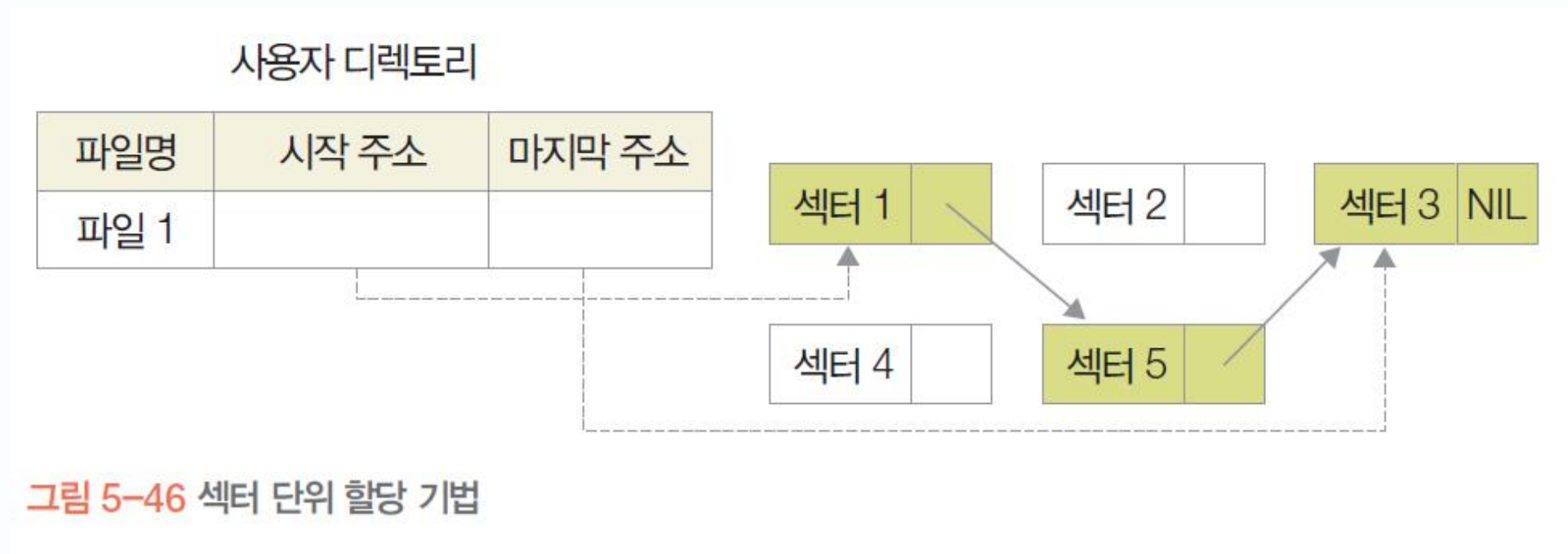
그림 5-45 디스크 공간의 연속 할당

3. 정보 관리

3) 파일의 공간 할당 기법

◆ 불연속 파일

- 섹터 단위 할당
 - 하나의 파일에 속한 다수의 섹터들이 연결 리스트로 구성
 - 각 섹터는 다른 섹터와의 연결을 위한 포인터를 가짐
 - 디렉터리는 각 파일의 시작 주소와 마지막 주소에 대한 포인터를 보유





요약

◆ CPU 스케줄링

- CPU를 언제, 어느 프로세스에 할당 할 것인지 결정하는 작업이다.

◆ 주기억장치 관리기법

- 단일 프로그래밍 기법
- 고정분할 다중 프로그래밍 기법
- 가변 분할 다중 프로그래밍 기법



요약

◆ 가상기억장치

- 프로그램의 일부를 주기억장치에 적재하고 나머지는 보조기억장치에 유지하여 프로그램 전체가 주기억장치에 존재하지 않아도 실행을 가능하게 한다.

◆ 파일 구조

- 파일을 구성하는 레코드가 보조기억장치에 저장되는 방식이다. 대표적인 파일구조는 순차파일, 색인 순차파일, 직접파일, 분할 파일 등이 있다.