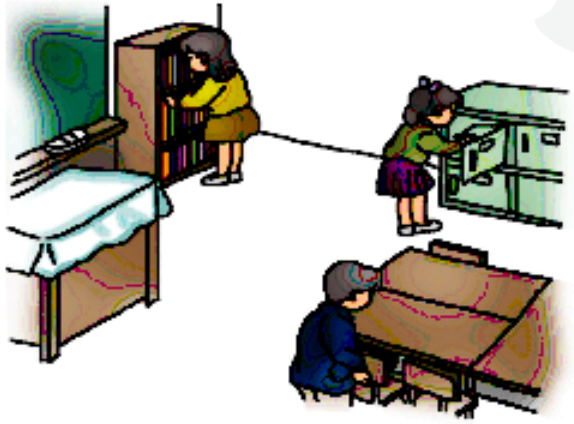
CSE2010 자료구조론

Week 14: Hashing 1

ICT융합학부 한진영

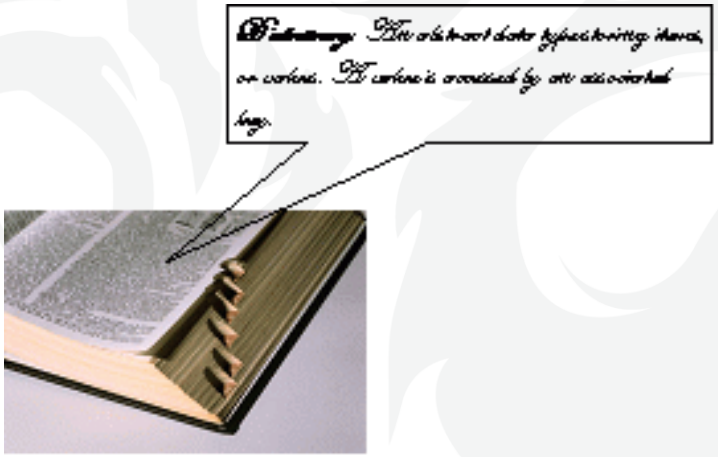
해싱이란?

- 대부분의 탐색 방법들은 키 값 비교로 탐색하고자 하는 항목에 접근
- 해싱(hashing)
  - 키 값에 대한 산술적 연산에 의해 테이블의 주소를 계산하여 항목에 접근
- 해시 테이블(hash table)
  - 키 값의 연산에 의해 직접 접근이 가능한 구조
- 해싱은 물건을 정리하는 것과 유사함



사전(Dictionary)

- 사전(dictionary)
  - 맵(map) 또는 테이블(table)로 불리움
  - 탐색 키와 관련된 값의 2가지 필드로 구성
    - 영어 단어나 사람의 이름 같은 탐색 키(search key)
    - 단어의 정의나 주소 또는 전화 번호 같은 탐색 키와 관련된 값(value)

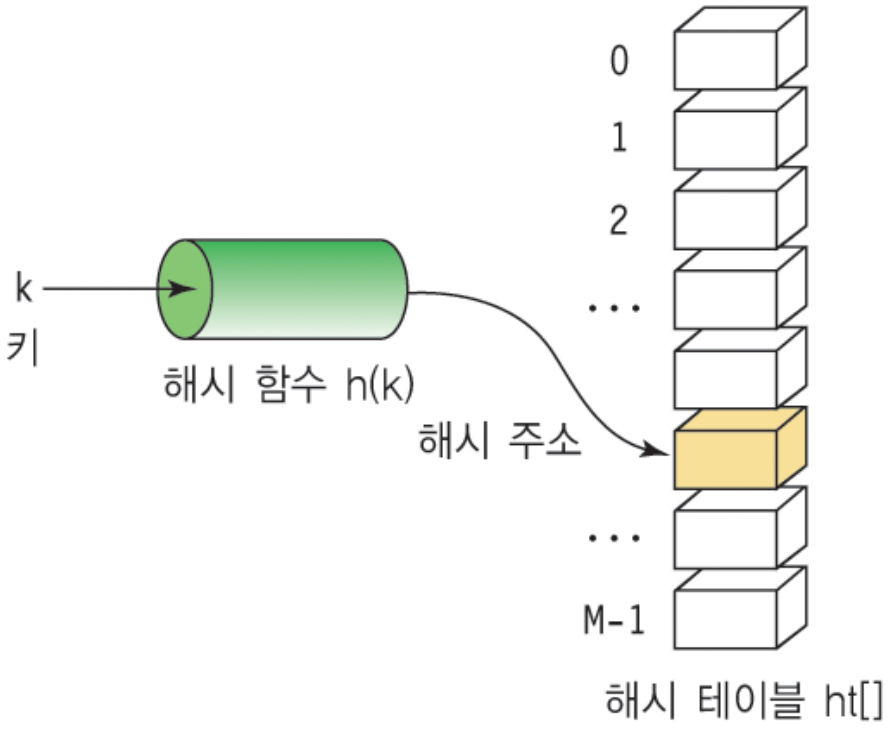


사전 ADT

·객체: 일련의 (key, value) 쌍의 집합
·연산:
- add(key, value) ::= (key, value)를 사전에 추가한다.
- delete(key) ::= key에 해당되는 (key, value)를 찾아서 삭제한다.
  관련된 value를 반환한다.만약 탐색이 실패하면 NULL를 반환한다.
- search(key) ::= key에 해당되는 value를 찾아서 반환한다.
  만약 탐색이 실패하면 NULL를 반환한다.

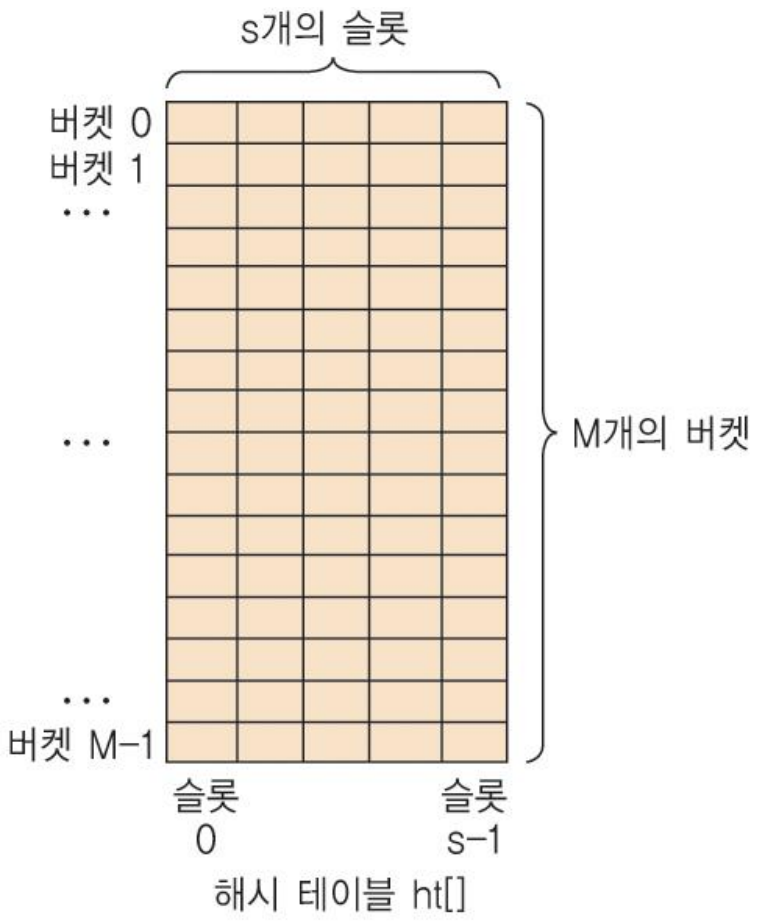
해싱의 구조

- 해시 함수(hash function)
  - 탐색키를 입력 받아 해시 주소(hash address) 생성
  - 해시 주소: 배열로 구현된 해시 테이블(hash table)의 인덱스



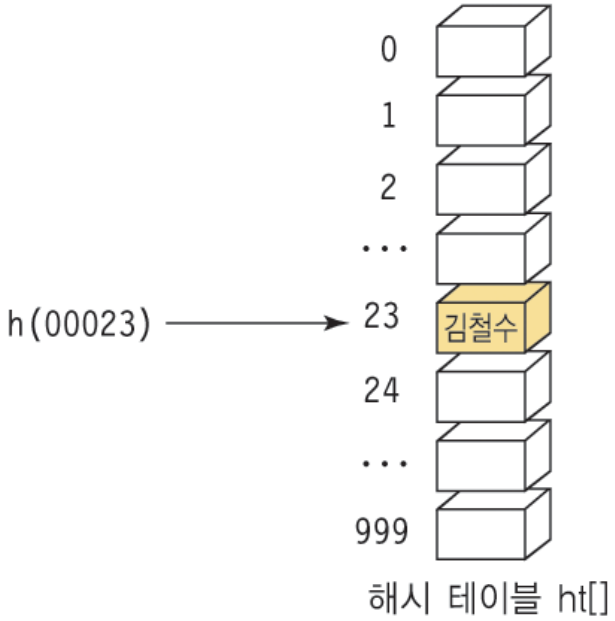
해시 테이블의 구조

- 해시테이블 ht
  - M개의 버켓(bucket)으로 구성된 테이블
  - ht[0], ht[1], ...,ht[M-1]의 원소를 가짐
  - 하나의 버켓에 s개의 슬롯(slot) 가능
- 충돌(collision)
  - 서로 다른 두 개의 탐색키 k1과 k2에 대하여 h(k1) = h(k2)인 경우
- 오버플로우(overflow)
  - 충돌이 버켓에 할당된 슬롯 수보다 많이 발생하는 것
  - 오버플로우 해결 방법 반드시 필요



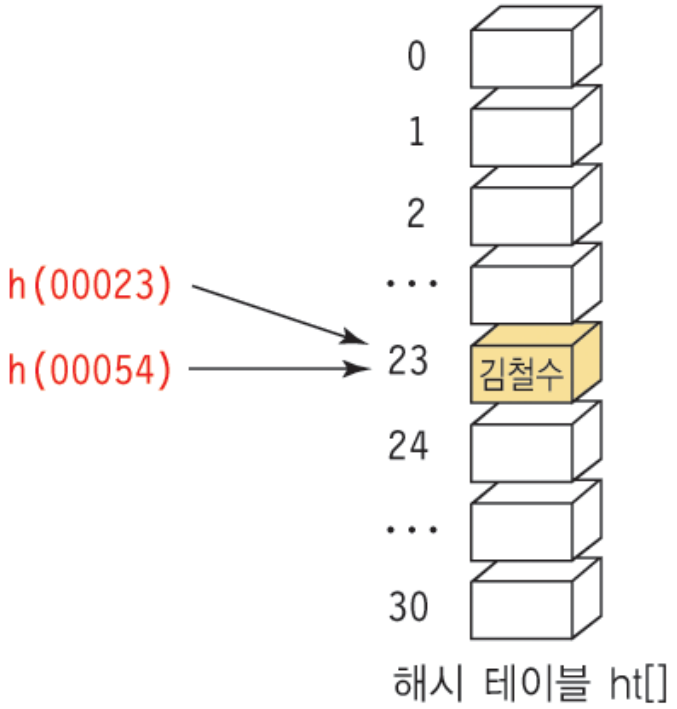
이상적인 해싱

- 학생 정보를 해싱으로 저장, 탐색해보자
  - 5자리 학번 중에 앞 2자리가 학과 번호, 뒤 3자리가 각 학과의 학생 번호
  - 같은 학과 학생들만 가정하면 뒤의 3자리만 사용해서 탐색 가능
  - 학번이 00023이라면 이 학생의 인적사항은 해시테이블 ht[23]에 저장
  - 만약 해시테이블이 1000개의 공간을 가지고 있다면 탐색 시간이 O(1) 되므로 이상적임



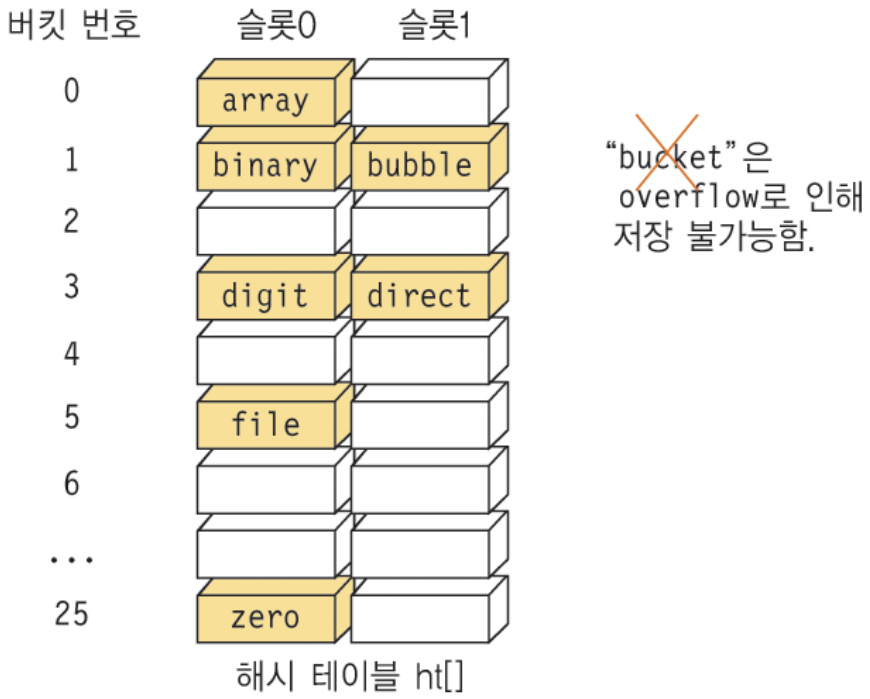
실제 해싱(1)

- 실제로는 해시테이블의 크기가 제한되므로, 존재 가능한 모든 키에 대해 저장 공간을 할당할 수 없음
- h(k)= k mod M 의 예에서 보듯이 필연적으로 충돌과 오버플로우 발생함



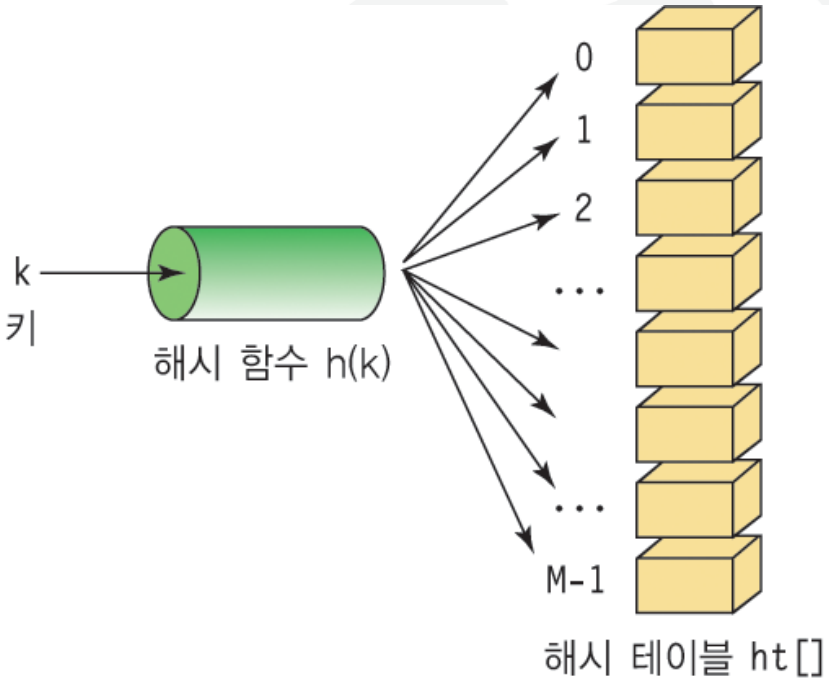
실제 해싱(2)

- 알파벳 문자열 키의 해시함수가 키의 첫 번째 문자의 순서
  - h("array")=1
  - h("binary")=2
- 입력데이터: array, binary, bubble, file, digit, direct, zero, bucket



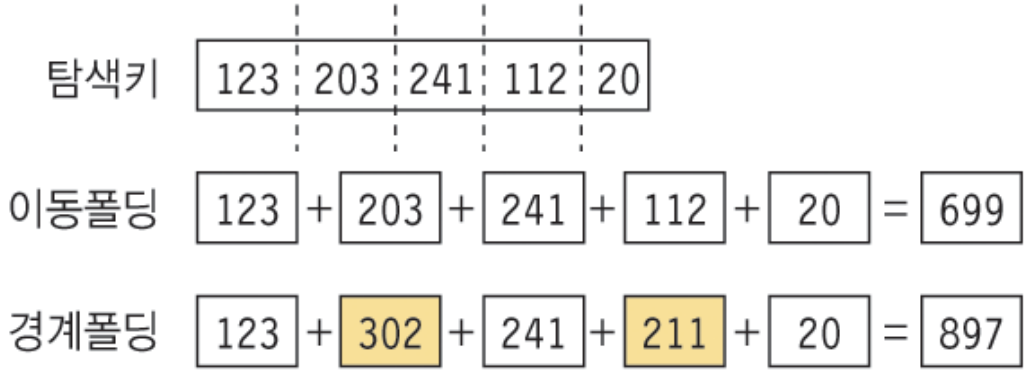
해시 함수(1)

- 좋은 해시 함수의 조건
  - 충돌이 적어야 함
  - 해시함수 값이 해시테이블의 주소 영역 내에서 고르게 분포되어야 함
  - 계산이 빨라야 함



해시함수(2)

- 제산 함수
  - h(k)=k mod M
  - 해시 테이블의 크기 M은 소수(prime number) 선택
- 폴딩 함수
  - 이동 폴딩(shift folding)과 경계 폴딩(boundary folding)



해시함수(3)

- 중간제곱 함수
  - 탐색키를 제곱한 다음, 중간의 몇 비트를 취해서 해시 주소 생성
- 비트추출 함수
  - 탐색키를 이진수로 간주하여 임의의 위치의 k개의 비트를 해시 주소로 사용
- 숫자 분석 방법
  - 키 중에서 편중되지 않는 수들을 해시테이블의 크기에 적합하게 조합하여 사용

Week 14: Hashing 1