CSE2010 자료구조론

Week 14: Hashing 2

ICT융합학부 한진영

충돌(Collision)

- 충돌(collision)
  - 서로 다른 탐색 키를 갖는 항목들이 같은 해시 주소를 가지는 현상
  - 충돌이 발생하면 해시 테이블에 항목 저장 불가능
  - 충돌을 효과적으로 해결하는 방법 반드시 필요

해시테이블

- 충돌해결책
  - 선형조사법: 충돌이 일어난 항목을 해시 테이블의 다른 위치에 저장
  - 체이닝: 각 버켓에 삽입과 삭제가 용이한 연결 리스트 할당

선형조사법(Linear Probing)

- 충돌이 ht[k]에서 발생했다면,
  - ht[k+1]이 비어 있는지 조사
  - 만약 비어있지 않다면 ht[k+2] 조사
  - 비어있는 공간이 나올 때까지 계속 조사
  - 테이블의 끝에 도달하게 되면 다시 테이블의 처음부터 조사
  - 조사를 시작했던 곳으로 다시 되돌아오게 되면 테이블이 가득 찬 것임
  - 조사되는 위치: h(k), h(k)+1, h(k)+2,...
- 군집화(clustering)과 결합(Coalescing) 문제 발생

선형조사법 예(1)

- 예: h(k)=k mod 7

1단계 (8) : h(8) = 8 mod 7 = 1(저장)
2단계 (1) : h(1) = 1 mod 7 = 1(충돌발생)
(h(1)+1) mod 7 = 2(저장)
3단계 (9) : h(9) = 9 mod 7 = 2(충돌발생)
(h(9)+1) mod 7 = 3(저장)
4단계 (6) : h(6) = 6 mod 7 = 6(저장)
5단계 (13) : h(13) = 13 mod 7 = 6(충돌 발생)
(h(13)+1) mod 7 = 0(저장)

	1단계	2단계	3단계	4단계	5단계
[0]					13
[1]	8	8	8	8	8
[2]		1	1	1	1
[3]			9	9	9
[4]					
[5]					
[6]				6	6

선형조사법 예(2)

- 예: “do”, “for”, “if”, “case”, “else”, “return”, “function”

탐색 키	덧셈식 변환 과정	덧셈 합계	해시 주소
do	100+111	211	3
for	102+111+114	327	2
if	105+102	207	12
case	99+97+115+101	412	9
else	101+108+115+101	425	9
return	114+101+116+117+114+110	672	9
function	102+117+110+99+116+105+111+110	870	12

버켓	1단계	2단계	3단계	4단계	5단계	6단계	7단계
[0]							function
[1]							
[2]		for	for	for	for	for	for
[3]	do	do	do	do	do	do	do
[4]							
[5]							
[6]							
[7]							
[8]							
[9]				case	case	case	case
[10]					else	else	else
[11]						return	return
[12]			if	if	if	if	if

이차 조사법(Quadratic Probing)

- 선형 조사법과 유사하지만, 다음 조사할 위치를 아래 식 사용
  - (h(k)+inc*inc) mod M
- 조사되는 위치는 다음과 같음
  - h(k), h(k)+1, h(k)+4,…
- 선형 조사법에서의 문제점인 군집과 결합 크게 완화 가능

이중해싱법(Double Hashing)

- 재해싱(rehashing)이라고도 함
  - 오버플로우가 발생하면 원 해시함수와 다른 별개의 해시 함수 사용
  - step=C-(k mod C)
  - h(k), h(k)+step, h(k)+2*step, …

- 예: 크기가 7인 해시테이블에서,
  - 첫 번째 해시 함수가 k mod 7
  - 오버플로우 발생시의 해시 함수는 step=5-(5 mod 5)
  - 입력 (8, 1, 9, 6, 13) 적용

이중해싱법 과정

1단계 (8) : h(8) = 8 mod 7 = 1(저장)
2단계 (1) : h(1) = 1 mod 7 = 1(충돌발생)
(h(1)+h'(1)) mod 7 = (1+5-(1 mod 5)) mod 7 = 5(저장)
3단계 (9) : h(9) = 9 mod 7 = 2(저장)
4단계 (6) : h(6) = 6 mod 7 = 6(저장)
5단계 (13) : h(13) = 13 mod 7 = 6(충돌 발생)
(h(13)+h'(13)) mod 7 = (6+5-(13 mod 5)) mod 7= 1(충돌발생)
(h(13)+2*h'(13)) mod 7 = (6+2*2) mod 7= 3(저장)

	1단계	2단계	3단계	4단계	5단계
[0]					
[1]	8	8	8	8	8
[2]			9	9	9
[3]					13
[4]					
[5]		1	1	1	1
[6]				6	6

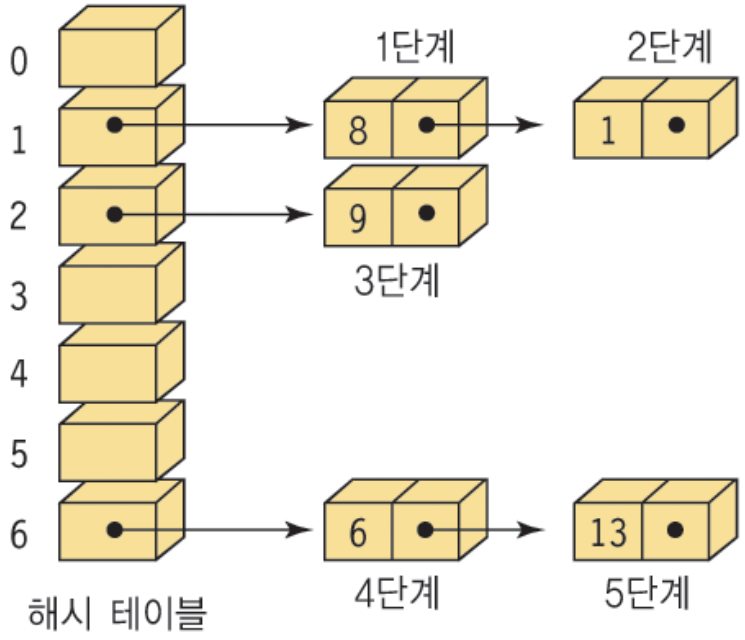
체이닝(Chaining)

- 오버플로우 문제를 연결 리스트로 해결
  - 각 버켓에 고정된 슬롯이 할당되어 있지 않음
  - 각 버켓에, 삽입과 삭제가 용이한 연결 리스트 할당
  - 버켓 내에서는 연결 리스트 순차 탐색

- 예: 크기가 7인 해시테이블에서
  - h(k)=k mod 7의 해시 함수 사용
  - 입력 (8, 1, 9, 6, 13) 적용

체이닝 과정

1단계 (8) : h(8) = 8 mod 7 = 1(저장)
2단계 (1) : h(1) = 1 mod 7 = 1(충돌발생->새로운 노드 생성 저장)
3단계 (9) : h(9) = 9 mod 7 = 2(저장)
4단계 (6) : h(6) = 6 mod 7 = 6(저장)
5단계 (13) : h(13) = 13 mod 7 = 6(충돌 발생->새로운 노드 생성 저장)



해싱의 성능분석: 선형조사법에서 비교연산 횟수

α	실패한 탐색	성공한 탐색
0.1	1.1	1.1
0.3	1.5	1.2
0.5	2.5	1.5
0.7	6.1	2.2
0.9	50.5	5.5

해싱의 성능분석: 체이닝에서 비교연산 횟수

α	실패한 탐색	성공한 탐색
0.1	0.1	1.1
0.3	0.3	1.2
0.5	0.5	1.3
0.7	0.7	1.4
0.9	0.9	1.5
1.3	1.3	1.7
1.5	1.5	1.8
2.0	2.0	2.0

Week 14: Hashing 2