자료구조응용

08. Hashing (20점)

2024.12.4.(수)

++ lms1 제출내용 : (1) 소스코드 (2) 문서파일 1개 (이름_학번.pdf/docx)

제출 기한: ~12/29(일)

* 문자열 데이터를 파일에서 입력받아 BST와 Hash 테이블에 각각 저장한 후, 삽입, 검색 성능 비교

실행 예시

```
========================================
   BST-Hashing 성능 비교
========================================
1. 사전 구성
2. 단어 삽입
3. 단어 검색
4. 삽입 성능 비교 (BST vs 해싱)
5. 검색 성능 비교 (BST vs 해싱)
6. 종료
----------------------------------------
선택:
```

```
선택: 1
파일 이름을 입력하세요: words.txt
BST와 해시 테이블에 삽입 완료

----------------------------------------
선택: 2
단어를 입력하세요: pineapple
- BST: S
- Hash: S

----------------------------------------
선택: 3
단어를 입력하세요: banana
- BST: S (비교 3회)
- Hash: S (인덱스 5, 비교 1회)
```

```
선택: 4
파일 이름을 입력하세요: insert.txt
파일 이름(삽입결과 출력)을 입력하세요: insertResult.txt

[삽입 성능]
데이터 개수 | BST          | 해싱
---------------------------------
10000       | 150 ms       | 100 ms


----------------------------------------
선택: 5
파일 이름을 입력하세요: search.txt
파일 이름(검색결과 출력)을 입력하세요:
searchResult.txt

[검색 성능]
데이터 개수 | BST          | 해싱
---------------------------------
10000       | 70 ms        | 30 ms
```

1. 입력 데이터

- 영어 단어 데이터셋 (10,000/50,000/100,000개)
- 입력 파일 형식 (예시: words.txt):

tangerine
cherry
papaya
date
kiwi
…

2. 데이터 저장 : BST, Hashing

- 해시 테이블 구현 : 동적 배열 사용
- 충돌 처리 방식 : Chaining 사용
- 해시 함수: Division Method 사용

3. 작업 유형 : (1) 사전 구성 (2) 단어 삽입, 검색 (3) 성능 비교 (BST, Hashing)

* 랜덤한 순서로 파일에 저장된 데이터셋을 읽어서 BST, Hash 테이블로 사전 구성
* 각 데이터 개수 10000, 50000, 100000개 각각에 대해서 테스트

4. 성능 비교

- 측정 대상: (1) 삽입 시간: BST vs Hashing (2) 검색 시간: BST vs Hashing.
- 측정 방법: 각 작업(삽입, 검색)에 걸리는 평균 시간을 측정

(삽입 시간 측정)

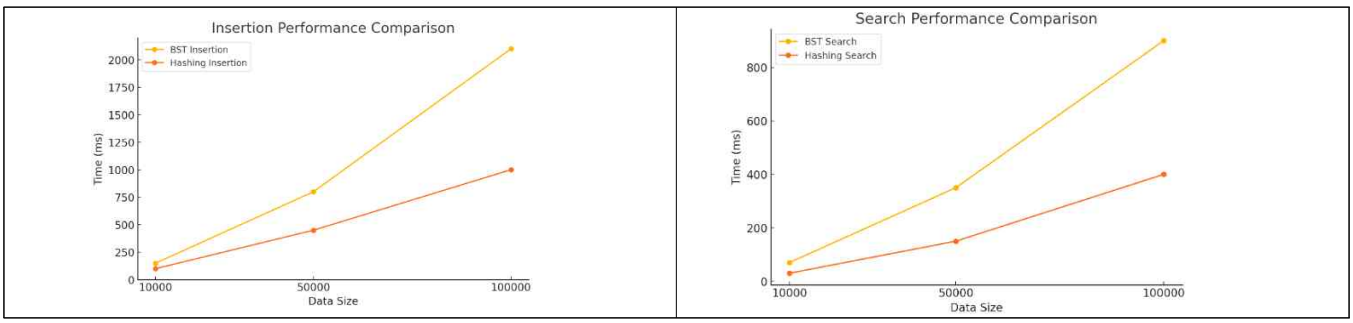
* insert.txt에 저장된 데이터 1,000개를 대상으로 삽입 수행
* 총 삽입 시간을 측정한 후, 평균 삽입 시간을 계산
* 예) 1000개의 단어를 삽입하는데 걸린 총 시간 / 1000 = 평균 삽입 시간(ms)

(검색 시간 측정)

* search.txt에 저장된 데이터 1,000개를 대상으로 검색 수행
* 총 검색 시간을 측정한 후, 평균 검색 시간 계산
* 예) 1000개의 검색 작업 총 수행 시간 / 1000 = 평균 검색 시간(ms)

5. 결과 시각화 (문서파일에 포함) : (1) 삽입 성능 (2) 검색 성능

(X축: 데이터 개수, Y축: 시간(ms)



* 문서파일 : (1) 실행결과 캡쳐 (2) 과제 핵심 요약 (3) 그래프, 결과 요약 등 포함