자료구조 프로젝트 리포트

2016/06/25

담당교수: 김종규

이름: 장용수

학과: 컴퓨터학과

학번: 2013210006

1. What data structure you chose and why?

User 목록을 저장하기 위해 Size가 100인 Hash Table을 만들고 같은 Hash Key를 가지는 User들을 RED-BLACK-TREE 로 구성하였습니다. (Chaining 기법에서 Linked List 대신 RED-BLACK-TREE를 사용함) 이때 Hash Table 에서 Collision 을 막기 위해 User의 ID를 이용하여 폴딩(Folding) 기법으로 Key 값을 만들었습니다. 또 RED-BLACK-TREE를 구성할 때 ID 값의 크고 작음을 기준으로 insert를 하였고, Fix up 과정을 통해 균형 잡힌 트리를 만들었습니다. Hash Table 과 RED-BLACK-TREE를 함께 적용한 이유는 ID를 입력 받았을 때 O(1) 시간으로 단번에 key값을 찾을 수 있고, O(logn)시간으로 user의 정보를 검색 할 수 있기 때문입니다. 또한 Hash Table 만을 이용했을 때보다 메모리를 적게 사용하고, RED-BLACK-TREE 만을 이용했을 때보다 빠르게 검색이 가능합니다. Friend.txt을 읽어 위에서 만든 구조에서 해당 User을 찾아 friend 목록에 추가하였습니다. Friend 목록은 Linked List로 구현하였습니다.

1. What is your expected performance?

User을 추가, 검색 하여 friend 목록, Profile 을 보는 것은 O(logn) 을 기대하였습니다. 균형 잡힌 트리로 구성하였기 때문에 기대에 충족합니다. 반면 아직 RED-BLACK-TREE 의 deletion 을 구현하지 못하여 일반 Binary-Tree 의 deletion 을 사용하였습니다. 이번 프로젝트와 같이 자료가 적을 때는 크게 무리가 없겠지만 대용량 파일을 다룰 때 균형 잡힌 트리를 구성하지 못한다면 속도에 무리가 있을 것이라고 생각합니다. Word를 읽어 들여 Linked-List 형태로 저장하였습니다. Top 5 most tweeted users를 보일 때 Linked-List 로 저장된 것을 배열로 만들어 ID를 기준으로 Heap Sort를 사용하여 정렬하였습니다. Heap Sort를 사용하면 O(nlogn) 시간으로 빨리 정렬 될 줄 알았지만 word의 양이 10만개가 넘어 정렬하는데 3~4분이 걸려 예상밖의 속도가 나왔습니다.

1. How would you improve the system in the future?

Word를 찾고 분석하는데 Linked-List를 사용한 결과 시간이 만족스럽지 못하였습니다. 새로운 구조를 생각해야 한다고 생각합니다. 아직 Word를 정렬하는데 어떤 기준을 잡을지 떠오르지 않습니다. 기준을 생각해 내면 균형 잡힌 트리로 구성하여 연산을 수행하는데 빠르도록 할 것입니다. RED-BLACK-TREE에서 RED-BLACK-Deletion 연산을 넣을 것입니다. Fix up 과정에서 Leaf node 를 처리하는 방안을 생각할 것입니다.