Київський національний університет імені Тараса Шевченка
Факультет комп`ютерних наук та кібернетики
Кафедра інтелектуальних програмних систем
Алгоритми та складність

Лабораторна робота №1
“Ідеальне хешування”
Виконав студент 2-го курсу
Групи К-28
Гуща Дмитро Сергійович

2020

Завдання

Реалізувати ідеальне хешування.

Теорія

Ідеальна хеш-функція – хеш функція, яка перетворює завчасно відому статичну множину ключів в діапазоні цілих чисел [0,n-1] без коллізій, тобто один ключ відповідає лише одному унікальному значенню.

Предметна область

Варіант 4
Предметна область: Учбовий відділ
Об’єкти: Групи, Студенти
Примітка: Маємо множину учбових груп. Кожна група містить в собі множину студентів

Алгоритм

1. Хешуємо елементи по групах(buckets). Таким чином ми уникнимо колізій.
2. Сортуємо групи від тих, з найбільшою кількістю елементів до тих, з найменшою кількістю.
3. Знаходимо модифікатор(salt value) для кожної групи, що коли кожен елемент з групи хешований, вони займають лише вільні місця.
4. Збережемо масив модифікаторів на кращі часи, він нам знадобиться під час пошуку інформації.
5. Якщо група має тільки 1 предмет, то замість того, щоб шукати значеннння модифікатора, ми можемо просто знайти вільний індекс і зберегти (індекс+1) в масив модифікаторів.

Коли ми обрахували ідеальне хешування, пошук будемо здійснювати наступним чином:

1. Хешуємо ключ і використовуємо цей хеш, щоб знайти який модифікатор використовувати.
2. Якщо значення модифікатора >= 0, то хешуємо ключ знову, використовуючи значення модифікатора.Результат цього хешування буде індексом в таблиці інформації(data table).
3. Якщо ж негативне, то беремо абсолютне значення і віднімаємо 1 – отримаємо індекс в таблиці інформації.
4. Можливо і таке, що ключ, який ми шукаємо, може бути не в таблиці, порівняємо цей ключ з ключем за цим індексом в таблиці. Якщо вони співпадають, повернемо інформацію по цьому індексу. В інакшому випадку ключа не було в таблиці.

Складність

Алгоритм працює за константний час O(1) в найгіршому випадку.

Мова програмування

C++

Модулі програми

student.h

```
class Student{}; //Класс опису студента
std::string getName(); // метод повертає ім’я студента
void getStudent();//метод виводить ID та ім'я студента в консоль
void setName(std::string name); //метод змінює ім'я студента
```

group.h

```
class Group {};
Group() : title("NULL"); //конструктор пустої групи
Group(std::string title); //конструктор з початковою назвою групи
Group(std::string title, Student* first_student); //конструктор з початковою назвою
групи та першим студентом
std::string getGroupTitle(); //модуль повертає назву групи
std::vector<Student*> getGroupStudents(); //модуль повертає множину студентів
void setGroupTitle(std::string title); //модуль змінює назву групи
void setGroupStudents(std::vector<Student*> students); //модуль змінює множину
студентів
void addStudent(Student* student); //додати нового студента
void printStudents(); //вивід у консоль усіх студентів групи
```

perfectHash.h

```
struct UniversalHash; // структуара яка описує функцію універсального хешування
class HashRow; //класс який описує рядок хеш-таблиці
class Core; //класс що описує поведінку в цілому всієї програми
void Core::run(); //метод який описує інтерактивний підхд до роботи програми
void Core::printGroups(); //метод що виводить усі групи з яких складається хеш-
таблиця
std::vector<std::string> Core::getGroupsTitles(int count); //метод повертає кількість
count назв груп
void Core::fillHashTable(HashRow* hashTable); //метод що заповнює даними хеш-таблицю
int Core::createHash(int id, int a, int b, int m); //метод що створює хеш-значення
для групи
void Core::printHashes(std::vector<std::vector<Group*>> hashesTable); //метод
виводить хеш-значення усієї таблиці
std::ostream& operator<<(std::ostream& os, HashRow hashRow); //перевантаження
оператору виводу
```

Інтерфейс користувача

Користувач вводить дані в консолі. Дані виводяться в текстовий файл

Тестовий приклад

Візьмемо таку множину груп. {К18,К19, К28, К29,IPS31}. Переведемо його до виду ASCII, отримаємо {754956,754957,755056,755057,7380835149}.

Звідки m = 6, p = 754923. Візьмемо a = 41, b = 460. Розрахуємо індекс комірки за формулою k = h(i) = ((a * i + b) mod p) mod m.

$$h(754956) = ((41 * 754956 + 460) mod\ 754923) mod\ 6 = 1$$

$$h(754957) = ((41 * 754957 + 460) mod\ 754923) mod\ 6 = 0$$

$$h(755056) = ((41 * 755056 + 460) mod\ 754923) mod\ 6 = 3$$

$$h(755057) = ((41 * 755057 + 460) mod\ 754923) mod\ 6 = 2$$

$$h(7380835149) = ((41 * 7380835149 + 460) mod\ 754923) mod\ 6 = 1$$

k	p	m	a	b	i	
0	754923	6	41	460	754957	
1					754956	7380835149
2					755057	
3					755056	

Для числа 1 отримали 2 початкові ключі (754956,7380835149), тому створюємо хеш-рядок розміром $m^2 = 9$, підбираємо нові a і b(a = 43321, b=2312). Знові рахуємо хеш-числа

$$h(754956) = ((1423 * 754956 + 2312) mod\ 754923) mod\ 9 = 5$$

$$h(7380835149) = ((1423 * 7380835149 + 2312) mod\ 754923) mod\ 9 = 8$$

k	p	m	a	b	i			
0	0	1	0	0	754957		NULL	
1	754923	9	1423	2312	754956	NULL	7380835149	NULL
2	0	1	0	0	755057		NULL	
3	0	1	0	0	755057		NULL	
4	0	0	0	0	755056		NULL	
5	0	0	0	0	NULL			
6	0	0	0	0	NULL			
7	0	0	0	0	NULL			
8	0	0	0	0	NULL			

Висновки

Реалізували ідеальне хешування для учбових груп. Ідеальне хешування є досить ефективним, до плюсів можна віднести те що при використанні ідеального хешування ми уникаєм колізій, але до мінусів можна віднести те, що множина ключів статична.

Література:

- https://neerc.ifmo.ru/wiki/index.php?title=%D0%A5%D0%B5%D1%88-%D1%82%D0%B0%D0%B1%D0%BB%D0%B8%D1%86%D0%B0#.D0.A5.D0.B5.D1.88.D0.B8.D1.80.D0.BE.D0.B2.D0.B0.D0.BD.D0.B8.D0.B5
- https://neerc.ifmo.ru/wiki/index.php?title=%D0%98%D0%B4%D0%B5%D0%B0%D0%BB%D1%8C%D0%BD%D0%BE%D0%B5_%D1%85%D0%B5%D1%88%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5
- https://habr.com/ru/post/254431/
- https://en.wikipedia.org/wiki/Perfect_hash_function