ITIVITI

LET IT TRADE

Алгоритмы и структуры данных

Косяков Михаил Сергеевич
к.т.н., доцент кафедры ВТ

УНИВЕРСИТЕТ ИТМО

Содержание курса

- Введение в теорию алгоритмов
- Алгоритмы сортировок
- **Структуры данных**
  - Линейные структуры
  - Бинарные деревья поиска
  - **Хеши и хеш-функции**
- Алгоритмы на графах
  - Обходы графов в ширину и глубину
  - Минимальные остовные деревья
  - Поиск кратчайших путей в графе

ITIVITI
LET IT TRADE
УНИВЕРСИТЕТ ИТМО

Мотивация

ITIVITI

LET IT TRADE

- Поиск с использованием индексирования по ключу
  - в отличие от поиска на основе операции сравнения
- Концептуально хеш-таблицу можно рассматривать как (неотсортированный) массив
  - непосредственное обращение к элементам по индексу
- Назначение: поддерживать динамический набор данных и эффективно выполнять следующие операции
  - поиск элемента
  - вставка нового элемента
  - удаление существующего элемента (по ключу, по указателю / handle)

УНИВЕРСИТЕТ ИТМО

Мотивация

- Ожидаемое константное время выполнения операций
  - для «правильных» реализаций
  - нет хороших гарантий для наихудшего случая
- Применение:
  - сопоставление объекта его идентификатору (телефонный справочник)
  - дедупликация (в потоке данных)
  - таблица символов при компиляции
  - фильтры сетевого трафика
  - и т.д. (задача о 2-сумме: есть ли такие A и B, что A+B=X)

LET IT TRADE

ITIVITI
УНИВЕРСИТЕТ ИТМО

Мотивация

- Таблица "Customers"

CustomerID	CustomerName	ContactName	Address	City	PostalCode	Country
1	Alfreds Futterkiste	Maria Anders	Obere Str. 57	Berlin	12209	Germany
2	Ana Trujillo Emparedados y helados	Ana Trujillo	Avda. de la Constitución 2222	México D.F.	05021	Mexico
3	Antonio Moreno Taquería	Antonio Moreno	Mataderos 2312	México D.F.	05023	Mexico
4	Around the Horn	Thomas Hardy	120 Hanover Sq.	London	WA1 1DP	UK
5	Berglunds snabbköp	Christina Berglund	Berguvsvägen 8	Luleå	S-958 22	Sweden
...	...	...	...	...	...	...

https://www.w3schools.com/sql/

LET IT TRADE
ITIVITI
УНИВЕРСИТЕТ ИТМО

Мотивация
Дедупликация

ITIVITI

- Пример DISTINCT (сравнить с DISTINCT при сортировке)

```
1.SELECT DISTINCT Country FROM Customers;
```

Number of Records: 21

Country
Germany
Mexico
UK
Sweden
France
Spain
Canada
Argentina
Switzerland
Brazil
…

LET IT TRADE

https://www.w3schools.com/sql/

УНИВЕРСИТЕТ ИТМО

Мотивация
Агрегация

ITIVITI

- Пример GROUP BY (сравнить с GROUP BY при сортировке)

```
1.SELECT COUNT(CustomerID), Country
2.FROM Customers
3.GROUP BY Country;
```

COUNT(CustomerID)	Country
11	Germany
5	Mexico
7	UK
2	Sweden
11	France
5	Spain
3	Canada
3	Argentina
…	…

LET IT TRADE

https://www.w3schools.com/sql/

УНИВЕРСИТЕТ ИТМО

HASH JOIN

ITIVITI

- Таблица "Customers"

CustomerID	CustomerName	ContactName	Country
1	Alfreds Futterkiste	Maria Anders	Germany
2	Ana Trujillo Emparedados y helados	Ana Trujillo	Mexico
3	Antonio Moreno Taquería	Antonio Moreno	Mexico
…	…	…	…

- Таблица "Orders"

OrderID	CustomerID	OrderDate
10308	2	1996-09-18
10309	37	1996-09-19
10310	77	1996-09-20
…	…	…

LET IT TRADE

https://www.w3schools.com/sql/

УНИВЕРСИТЕТ ИТМО

HASH JOIN

- Пример (INNER) JOIN

```
1.SELECT Orders.OrderID, Customers.CustomerName, Orders.OrderDate
2.FROM Orders
3.INNER JOIN Customers
4.ON Orders.CustomerID=Customers.CustomerID;
```

OrderID	CustomerName	OrderDate
10308	Ana Trujillo Emparedados y helados	1996-09-18
10309	Hungry Owl All-Night Grocers	1996-09-19
10310	The Big Cheese	1996-09-20
10311	Du monde entier	1996-09-20
10312	Die Wandernde Kuh	1996-09-23
…	…	…

- Can only be used for equi-joins on the complete join key

https://www.w3schools.com/sql/

LET IT TRADE
ITIVITI
УНИВЕРСИТЕТ ИТМО

Прямая адресация

key	0	1	2	3	4	…	M -1	M
value	v_0	v_1	*empty_val*	v_3	v_4	…	v_{M-1}	v_M

- Поиск по ключу: возвращаем T[k]
- Вставка элемента с ключом k и значением v : T[k] = v
- Удаление по ключу: T[k] = empty_val
- Проблемы:
  - M может быть очень большим, при этом в таблице обычно хранится лишь небольшое количество ключей
  - нецелочисленные ключи (двоичное представление?)

LET IT TRADE
ITIVITI
УНИВЕРСИТЕТ ИТМО

Хеш-таблица

- U – множество всех возможных ключей
- K – множество всех хранимых ключей, $|K| << |U|$, $|K| = N$
- M – размер хеш-таблицы, число всех ее адресов (ячеек или «бакетов»), в т.ч. пустых и с удаленными элементами
- $h(k)$ – хеш-функция, которая сопоставляет каждому ключу из U число из множества $\{0, 1, \ldots, M-1\}$
- Элементы с ключом k сохраняются (хешируются) в таблице по адресу $h(k)$

index	0	…	$h(k_1)$	$h(k_9)$	$h(k_7)$	$h(k_3)$	…	M
value	…	…	v_1	*empty_val*	v_7	*del_val*	…	…

LET IT TRADE
ITIVITI
УНИВЕРСИТЕТ ИТМО

Хеш-функции

- Если размер таблицы *M*, то диапазон значений хеш-функции должен быть от 0 до *M*-1
- Мультипликативная хеш-функция:
  - Преобразуем ключи в числа с плавающей точкой в диапазоне 0..1 и умножаем на *M*
  - Старшие разряды ключа определяют хеш-значение
  - Ключи должны быть распределены по диапазону равномерно
- Модульная хеш-функция:
  - Выбираем в качестве размера таблицы *M* простое число
  - Преобразуем ключи в целые числа и вычисляем остаток от деления на *M*

LET IT TRADE

ITIVITI
УНИВЕРСИТЕТ ИТМО

Хеш-функции

- $U = \{0, 1, 2, \ldots, 2^{64}-1\}$
- $M = 7$
- $N = 5$
- $h(k) = k \bmod M$
- Элементы с ключом k сохраняются в таблице по адресу $h(k)$
- Вставим последовательность ключей 1, 2, 5, 7, 14 (key = value)

index	0	1	2	3	4	5	6
value							

LET IT TRADE
ITIVITI
УНИВЕРСИТЕТ ИТМО

Хеш-функции

- $U = \{0, 1, 2, \ldots, 2^{64}-1\}$
- $M = 7$
- $N = 5$
- $h(k) = k \bmod M$
- Элементы с ключом k сохраняются в таблице по адресу $h(k)$
- Вставим последовательность ключей 1, 2, 5, 7, 14 (key = value)

index	0	1	2	3	4	5	6
value	7, 14	1	2			5	

LET IT TRADE
ITIVITI
УНИВЕРСИТЕТ ИТМО

Коллизии

- Два различных ключа могут иметь одинаковое хеш-значение, т.е. k_1 != k_2 но $h(k_1) == h(k_2)$
  - Такой случай называется **коллизией**
- На футбольном поле игроки двух команд по 11 человек, включая вратарей. Игру судит 1 арбитр. Какова вероятность того, что у двух людей на поле день рождения в один день?
  - 0.1
  - 0.2
  - 0.5
  - 0.9

LET IT TRADE

ITIVITI
УНИВЕРСИТЕТ ИТМО

Коллизии

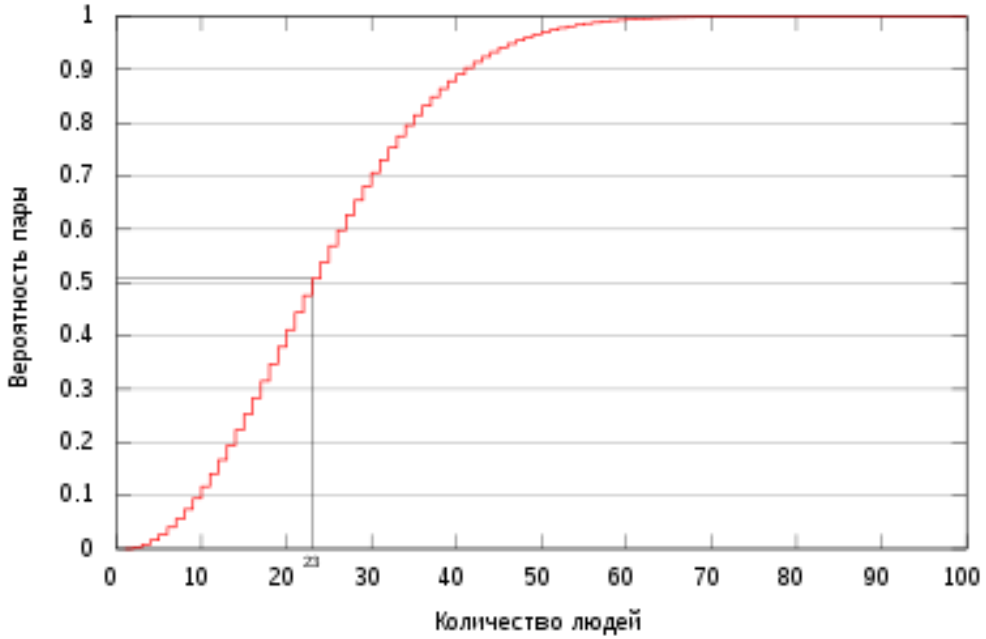
ITIVITI

- Сколько нужно собрать людей в одной комнате, чтобы у двоих из них день рождения был в один день с вероятностью 0.99 ? … с вероятностью 1 ?
  - 23
  - 57
  - 184
  - 367
- Принцип Дирихле:
  - Если кролики рассажены в клетки, причём число кроликов больше числа клеток, то хотя бы в одной из клеток находится более одного кролика

LET IT TRADE

УНИВЕРСИТЕТ ИТМО

Парадокс дней рождения



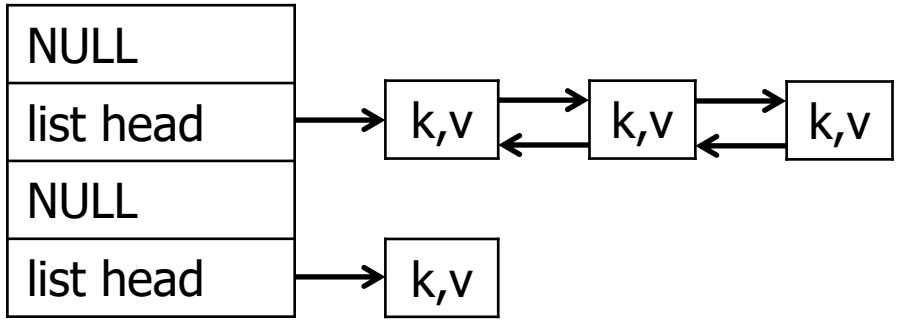
- $Pr = 1 - \frac{N-1}{N} \cdot \frac{N-2}{N} \cdot \ldots \cdot \frac{N-(k-2)}{N} \cdot \frac{N-(k-1)}{N} \approx 1 - e^{\frac{-k(k-1)}{2N}}$
- Для малых X: $1 - e^{-X} \approx X$, $Pr \sim \frac{k^2}{2N}$

https://ru.wikipedia.org/wiki/Парадокс_дней_рождения

LET IT TRADE
ITIVITI
УНИВЕРСИТЕТ ИТМО

Метод цепочек

- Ячейка T[$h(k)$] содержит (двусвязный) список всех элементов с одинаковым хеш-значением
  - Каждый элемент списка хранит и ключ, и значение
- Операции с хеш-таблицей ~ операции на списке
  - Вместо одного списка поддерживаются M списков



Метод цепочек

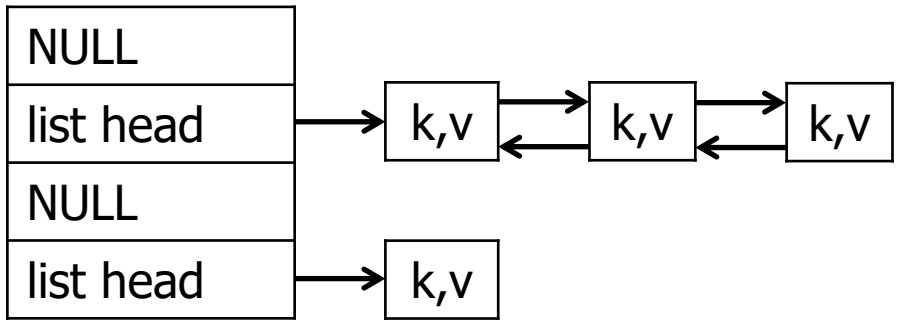
- Стоимость операции вставки (при условии $h(k) = O(1)$):
  - неотсортированные списки
  - отсортированные списки
- Поддержка дубликатов?
- Неотсортированные списки
  - легко искать / удалять последние вставленные элементы
- Множество операций вставки / удаления / поиска и однократный вывод в отсортированном виде
  - вывод неотсортированных списков: $O(N \log N)$
  - вывод отсортированных списков: $O(N \log M)$

LET IT TRADE

ITIVITI
УНИВЕРСИТЕТ ИТМО

Метод цепочек

- От чего зависит время поиска / удаления в такой хеш-таблице?



LET IT TRADE
ITIVITI
УНИВЕРСИТЕТ ИТМО

Метод цепочек

- От чего зависит время поиска / удаления в такой хеш-таблице?
- От длины списка:
  - числа коллизий, т.е. хеш-функции
  - размера таблицы *M* и числа хранимых в ней элементов *N*
- Если хеш-функция равномерно перемешивает ключи, то поиск в каждом из *M* списков равновероятен
  - средняя длина списков *N* / *M*
  - стоимость операций уменьшается (в среднем) в *M* раз
- «Допустимая» средняя длина списка ≈ от 5 до 10 элементов

LET IT TRADE

ITIVITI
УНИВЕРСИТЕТ ИТМО

Метод цепочек

- Для таблицы с M ячейками и N элементами определим коэффициент заполнения $\alpha = N / M$
  - В худшем случае время поиска составляет $\Theta(N)$
  - В хеш-таблице с разрешением коллизий методом цепочек время неудачного поиска в предположении равномерного хеширования составляет $\Theta(\alpha + 1)$
- Важно поддерживать α константным
  (для std::unordered_map default max_load_factor = 1.0)
- Важно равномерно перемешивать ключи: хеширование ключа в случайную ячейку
  - Но где потом искать данный ключ?

LET IT TRADE

ITIVITI
УНИВЕРСИТЕТ ИТМО

Хеш-функции

- «Идеальная» хеш-функция:
  - Быстро вычисляется за константное время
  - Равномерно распределяет любой (!) входной набор ключей
  - Проблема: такой функции не существует! (принцип Дирихле)
- Примеры плохих хеш-функций:
- Ключи – телефонные номера, $|U|=10^{10}$, $|K|=10^3$
  - Первые три цифры?
  - Последние три цифры?
- Ключи – адреса объектов в памяти (почти всегда четны)
  - Последние биты: $h(k) = k \bmod$ (степень двойки) ?

LET IT TRADE

ITIVITI
УНИВЕРСИТЕТ ИТМО

Модульные хеш-функции

ITIVITI

- Преобразуем ключи в целые числа и вычисляем остаток от деления на M
- Целочисленные представления вставляемых ключей и размер таблицы M не должны иметь общих делителей
  - M – простое число
- Целочисленное представление для составных ключи (строки):
- Misha: $(77 \cdot 128^4 + 105 \cdot 128^3 + 115 \cdot 128^2 + 104 \cdot 128^1 + 97 \cdot 128^0) \% M$
  - Слишком много разрядов для арифметических операций
- Схема Горнера
  - Misha: $((((77 \cdot 128 \% M + 105) \cdot 128 \% M + 115) \cdot 128 \% M + 104) \cdot 128 \% M + 97) \% M$, т.к. $(ax + b) \% M = ((ax \% M) + b) \% M$

LET IT TRADE

УНИВЕРСИТЕТ ИТМО

Хеш-функции
Патологические данные

- $U = \{0, 1, 2, \ldots, 2^{64}-1\}$
- $M = 7$
- $N = 4$
- $h(k) = k \bmod M$
- Элементы с ключом k сохраняются в таблице по адресу $h(k)$
- Вставим последовательность ключей 1, 8, 15, 22 (key = value)

index	0	1	2	3	4	5	6
value							

LET IT TRADE
ITIVITI
УНИВЕРСИТЕТ ИТМО

Хеш-функции
Патологические данные

- $U = \{0, 1, 2, \ldots, 2^{64}-1\}$
- $M = 7$
- $N = 4$
- $h(k) = k \bmod M$
- Элементы с ключом k сохраняются в таблице по адресу $h(k)$
- Вставим последовательность ключей 1, 8, 15, 22 (key = value)

index	0	1	2	3	4	5	6
value		1,8,15,22					

LET IT TRADE
ITIVITI
УНИВЕРСИТЕТ ИТМО

Универсальное хеширование

ITIVITI

- (Любая) фиксированная хеш-функция уязвима
  - Открытый исходный код
- Давайте использовать рандомизацию!
- Построим семейство хеш-функций, такое что для любого входного набора ключей, «почти все» функции будут распределять ключи «достаточно» равномерно
- Будем выбирать хеш-функцию случайным образом из этого семейства в начале работы программы
  - Одни и те же входные данные не будут постоянно давать наихудшее поведение
  - Очень низкая вероятность выбора «плохой» хеш-функции
  - Мат. ожидание операции поиска $O(1)$

LET IT TRADE

Универсальное хеширование

ITIVITI

- STL: Hash functions are only required to produce the same result for the same input within a single execution of a program; this allows salted hashes that prevent collision denial-of-service attacks
- Abseil: The hash codes computed by absl::Hash are not guaranteed to be stable across different runs of your program. In fact, in the usual case it randomly seeds itself at program startup
  - Набор ключей по разному распределится по ячейкам копий хеш таблиц в разных процессах (источник недетерминизма для RSM)
  - Порядок итерации по копиям хеш таблиц будет давать разные результаты
- Cryptographic hashing with salt: объединим ключ k вместе со случайным значением S (salt) и вычислим $h(k, S)$
  - Пароли хранятся как пары <salt>:<hash>

https://en.m.wikipedia.org/wiki/Salt_(cryptography)

LET IT TRADE

УНИВЕРСИТЕТ ИТМО

Открытая адресация

- Не используем память под указатели: в каждой ячейке T хранится один элемент
- Для разрешения коллизий полагаемся на пустые ячейки (размер таблицы $M > N$)
- Хеш-функция теперь определяет последовательность проб (исследование) внутри массива T
  - При наличии конфликта (при поиске / вставке) проверяем следующую ячейку
  - Пока не встретим пустую ячейку
- Самый простой пример серии проб – линейное исследование: проверяются индексы $h(k)$, $h(k) + 1$, $h(k) + 2$, …, 0, 1, $h(k) - 1$

ITIVITI
LET IT TRADE
УНИВЕРСИТЕТ ИТМО

Открытая адресация

ITIVITI

- При проверке ячейки T могут быть следующие ситуации:
  - Ячейка содержит элемент, ключ которого совпадает с искомым / вставляемым
  - Ячейка содержит элемент, ключ которого не совпадает с искомым / вставляемым
  - Ячейка пуста
  - Ячейка содержит удаленный ключ
- Коэффициент заполнения $\alpha = N / M$
  - Метод цепочек – среднее число элементов в одном списке; в общем случае $\alpha \geq 1$
  - Открытая адресация – доля занятых ячеек в таблице; всегда должно быть $\alpha \leq 1$

LET IT TRADE

УНИВЕРСИТЕТ ИТМО

Открытая адресация

- $U = \{0, 1, 2, \ldots, 2^{64}-1\}$
- $M = 7,\ \ h(k) = k \bmod M$
- Линейное исследование
- Вставим последовательность ключей 0, 2, 4, 6

index	0	1	2	3	4	5	6
value	e	e	e	e	e	e	e

- Вставим последовательность ключей 1, 8, 15, 2

index	0	1	2	3	4	5	6
value	e	e	e	e	e	e	e

ITIVITI
LET IT TRADE
УНИВЕРСИТЕТ ИТМО

Открытая адресация

- $U = \{0, 1, 2, \ldots, 2^{64}-1\}$
- $M = 7,\ \ h(k) = k \bmod M$
- Линейное исследование
- Вставим последовательность ключей 0, 2, 4, 6

index	0	1	2	3	4	5	6
value	0	e	2	e	4	e	6

- Вставим последовательность ключей 1, 8, 15, 2

index	0	1	2	3	4	5	6
value	e	1	8	15	2	e	e

ITIVITI
LET IT TRADE
УНИВЕРСИТЕТ ИТМО

Открытая адресация

ITIVITI

- В методе цепочек поиск ключа осуществлялся среди ключей с таким же хеш-значением
- В открытой адресации обычно исследуются и другие ключи
  - Вставка ключа с один хеш-значением может повлиять на поиск ключей с другим хеш-значением
- Возникает проблема кластеризации: создание длинных последовательностей занятых ячеек
  - Из элементов, имеющих разные хеш-значения
- Линейное исследование приводит к наиболее сильной кластеризации

LET IT TRADE

Удаление элементов

- Удалим элемент с ключом 8

index	0	1	2	3	4	5	6
value	e	1	8	15	2	e	e

- Нет никаких сложностей в методе цепочек
- В открытой адресации «нельзя просто так взять и удалить элемент»
  - Перехеширование всех элементов между удаленным элементом и следующей пустой ячейкой
  - Замена удаленного ключа служебным (ячейка может быть переиспользована при вставке, но считается занятой при поиске)

LET IT TRADE
ITIVITI
УНИВЕРСИТЕТ ИТМО

Удаление элементов

- Удалим элемент с ключом 8

index	0	1	2	3	4	5	6
value	e	1	d	15	2	e	e

- Каждый ячейка либо содержит элемент, либо пуста, либо содержит удаленный ключ. Как хранить эту информацию?
- Какие проблемы возникают при частой вставке / удалении элементов?
  - Ячейки с удаленными ключами считаются занятыми при поиске
  - Кластеризация при использовании открытой адресации

LET IT TRADE

ITIVITI
УНИВЕРСИТЕТ ИТМО

Удаление элементов

ITIVITI

- Кластеризация при использовании открытой адресации
- Google dense hash table
  - не учитывает число проб для решения об увеличении размера таблицы; опирается только на $\alpha = N / M$

	#	50%	95%	99.9%
limit_check_start – limit_check_stop	1000	191 µs	216 µs	263 µs

- std::unordered_map

	#	50%	95%	99.9%
limit_check_start – limit_check_stop	1000	5 µs	9 µs	17 µs

LET IT TRADE

Квадратичное исследование

- $h(k, i) = (h(k) + c_1 \cdot i + c_2 \cdot i^2) \bmod M$,
- i – шаг исследования $\{0, 1, \ldots, M - 1\}$, c_1 и c_2 – константы
- Пример квадратичного исследования:
  - Google dense hash table использует последовательность треугольных чисел {0, 1, 3, 6, 10, 15, 21, …}
- Кластеризация мягче, чем при линейном исследовании
- Локальность проб для линеек кеша процессора
- НО! Последовательность проб определяется начальной позицией $h(k)$
  - всего M различных последовательностей
  - ключи с $h(k_1) == h(k_2)$ имеют одинаковые последовательности

https://ru.wikipedia.org/wiki/Треугольное_число

LET IT TRADE
ITIVITI
УНИВЕРСИТЕТ ИТМО

Двойное хеширование

- $h(k, i) = (h(k) + i \cdot h_2(k)) \bmod M$,
- i – шаг исследования $\{0, 1, \ldots, M - 1\}$, $h_2(k)$ – хеш-функция
- Последовательность проб определяется ключом k
- В сравнении с линейным исследованием:
  - Позволяет иметь бОльший α для такого же среднего числа проб при поиске
  - Не позволяет удалять элемент путем перехеширования других элементов (стоящих справа): различные последовательности проб могут проходить через удаляемый элемент
  - Удаление только через служебный ключ
- Все методы должны гарантировать проверку всех M ячеек хеш-таблицы

ITIVITI
LET IT TRADE
УНИВЕРСИТЕТ ИТМО

Открытая адресация

- При разрешении коллизий на основе цепочек при увеличении α увеличивается время поиска
- При открытой адресации таблица может заполниться
- Пусть каждая из $M!$ последовательностей проб равновероятна, тогда среднее число проб при поиске отсутствующего элемента ~ $1/(1-\alpha)$
  - Вероятность попасть в пустую ячейку на первой пробе $(1-\alpha)$, на второй: $\alpha(1-\alpha)$, на третьей: $\alpha^2(1-\alpha)$, …
  - Среднее число проб, чтобы добраться до пустой ячейки: $1\cdot(1-\alpha) + 2\cdot\alpha(1-\alpha) + 3\cdot\alpha^2(1-\alpha) + \ldots = (1-\alpha)\cdot[1 + 2\alpha + 3\alpha^2 + \ldots] = 1/(1-\alpha)$
- Для линейного исследования (Д. Кнут) ~ $1/(1-\alpha)^2$
- Важно поддерживать α константным!

LET IT TRADE
ITIVITI
УНИВЕРСИТЕТ ИТМО

Увеличение размера таблицы

ITIVITI

- Обычно α поддерживается в пределах 0.125-0.5, после чего изменяется размер таблицы
  - Таблицу увеличиваем вдвое при α = 0.5: α становится 0.25
  - Таблицу уменьшаем вдвое при α = 0.125: α становится 0.25
- Можно увеличивать размер таблицы при превышении ограничения на число проб (Robin Hood hashing)
- Какие проблемы возникают при увеличении размера таблицы?
  - Выделение памяти и копирование всех элементов
  - Перехеширование всех элементов
- Амортизированная стоимость вставки аналогична std::vector

LET IT TRADE

Хеш-таблица vs сбалансированное БДП

ITIVITI

- В хеш-таблице элементы хранятся в неупорядоченном виде
- Время обхода хеш-таблицы обычно больше, чем дерева (особенно при открытой адресации)
- Сбалансированное дерево дает гарантию $O(\log N)$ в худшем случае
- Добавление элементов в дерево никогда не приводит к полному копированию дерева и другим «тяжелым» операциям, кроме выделения памяти для нового узла
- Для любой хеш-функции существует последовательность паталогических данных

LET IT TRADE

УНИВЕРСИТЕТ ИТМО

ITIVITI

LET IT TRADE

Спасибо за внимание!

УНИВЕРСИТЕТ ИТМО