ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ
ФГАОУ ВО НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ
«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»

Факультет компьютерных наук
Образовательная программа «Прикладная математика и информатика»

УДК: 004.9

Отчет об исследовательском проекте

на тему: Обратимые фильтры Блума и сравнение геномов

(промежуточный, этап 1)

Выполнил:
Студент группы БПМИ209  М.М.Марченко
Подпись И.О.Фамилия

17.02.2022
Дата

Принял:
Руководитель проекта Григорий Аронович Кучеров
Имя, Отчество, Фамилия

Ведущий научный сотрудник
Должность, ученое звание

CNRS, France
Место работы (Компания или подразделение НИУ ВШЭ)

Дата проверки 2022 Оценка (по 10-ти бальной шкале) Подпись

Москва 2022

Содержание

1 Введение

Фильтр Блума — пожалуй, самая известная вероятностная структура данных. Фильтр Блума входит в программу большинства курсов по алгоритмам и, как известно, уже много где применен. Одна из вариаций фильтра Блума - менее известный обратимый фильтр Блума (Goodrich и Mitzenmacher [1]), занимает больше памяти, однако обладает сильно расширенной функциональностью.

Одним из преимуществ обратимого фильтра Блума является при наличии двух наборов данных A и B, отличающихся на t, сжимать их до O(t) и восстанавливать (с помощью либо A, либо B) с низкой вероятностью ошибки.

Геном человека занимает большое количество памяти, однако, как известно, может отличаться лишь в 0.1% мест. Данная работа ставит целью оптимизировать использование памяти для хранения генома, а также поэксперементировать с сравнением фильтров с запакованными геномами и восстановлению разницы между ними.

2 Цели

1. Реализовать обратимый фильтр Блума на C++.
2. Проинициализировать результат данными генома.
3. Оптимизировать обратимый фильтр Блума по памяти и вероятности ошибки конкретно для случая хранения генома.
4. В случае удовлетворительных результатов по оптимизации создать удобный интерфейс для запаковки и распаковки генома.
5. Поэксперементировать с несколькими фильтрами, содержащими каждый свой геном.
6. Попробовать восстановить разницу между двумя геномами запакованными в фильтры.

3 Фильтры Блума

3.1 Фильтр Блума

Обычный фильтр Блума представляет собой битовый массив и набор из k хеш-функций. По фильтру невозможно восстановить множество, хранящееся в нем, невозможно удалить элемент. Поддерживает операции:

1. insert(x) — метод добавляет элемент x в множество, делая 1 биты соответствующие каждой из k хеш-функций от x.
2. query(x) — отвечает на запрос есть ли элемент x в множестве. Метод проверяет каждый бит фильтра соответствующий каждой хеш-функции от x. Если хотя бы один из них равен 0 - элемента точно нет в множестве. Иначе дается положительный ответ (с маленькой вероятностью ложноположительный).

3.2 Обратимый фильтр Блума

В отличие от обычного фильтра Блума, обратимый хранит пары из ключей и значений (в этом варианте ключи разные). Обратимый фильтр Блума состоит из k хеш-функций и трех массивов: count (количество элементов в текущей ячейке), keyxor (xor (или сумму) ключей, лежащих в данной ячейке) и valuexor (xor (или сумму) значений, лежащих в данной ячейке). Обратимый фильтр Блума поддерживает операции:

1. insert(key, value) — метод вставки пары в множество. Для каждой хеш-функции от key к соответствующей ячейке count прибавляется 1, ячейка keyxor ксорится с key, ячейка valuexor с value.

2. remove(key, value) — метод удаления пары из множества. Для каждой хеш-функции от key от соответствующей ячейки count отнимается 1, ячейка keyxor ксорится с key, ячейка valuexor с value (поскольку xor ассоциативен, коммутативен и (a xor a = 0))

3. get(key) — метод получения значения по ключу. Если хотя бы одна из ячеек, соответствующих хеш-функциям от value, пустая (count = 0), то в множестве нет пары для key. Иначе, если есть ячейка с count = 1, то пара key содержится в соответствующей ячейке valuexor.

4. list entries() — метод, очищающий фильтр и выводящий все пары, содержащиеся в нем. Метод работает следующим образом: ищет по массиву count ячейки равные 1, выводит соответствующую пару и удаляет ее из фильтра. Операция повторяется пока не в фильтре не очистится (с маленькой вероятностью в непустом фильтре может остаться ни одной единицы - тогда вывод метода будет неполным).

3.3 Устойчивость к некорректной вставке и удалению

Для случая, когда при использовании фильтра удаляют или вставляют несколько пар с одинаковым ключом и разными значениями в статье (Goodrich и Mitzenmacher [1]) предлогается хранить дополнительный массив hashvaluesum, содержащий сумму хешей от значений, хранящихся в текущей ячейке.

4 Сжатие генома в фильтр Блума

Геномы хранятся как пары: название локуса (строка) и значение нуклеотида в локусе (число). Геном состоит из 600 млн нуклеотидов, при этом геномы двух людей различаются в лишь около 3 млн мест. Места различий могут быть разными для разных пар геномов, поэтому задача нахождения различий и задача сжатия генома не тривиальны.

4.1 Сжатие и восстановление геномов

Рассмотрим следующую задачу из статьи (Goodrich и Mitzenmacher [1]):

Алиса (А) хочет отправить Бобу (Б) базу данных закодированную в виде набора из пар ключей и значений (при этом все ключи разные). При этом у Б имеется старая версия данных и версии А и Б могут отличаться лишь на t (t значительно меньше размера всех данных). А создает фильтр Блума размера порядка O(t), вставляет (метод insert) туда все пары из данных А и отправляет получившийся фильтр Б. Б удаляет (метод remove) из полученного фильтра все пары из данных Б. Теперь в фильтре находятся только пары отличающиеся у А и Б. У пар А значение в массиве count равно 1, у пар Б значение в массиве count равно -1. Теперь Боб выводит (метод list entries) все оставшиеся пары.

Проблема со сжатием генома таким образом заключается в том, что ключи — названия локусов в нашем случае, совпадают у А и у Б. Поэтому при сжатии генома будет использована версия фильтра Блума устойчивая к нескольким парам с одинаковым ключом (то есть версия с дополнительным массивом G).

4.2 Сравнение геномов

Известно, что при наличие двух фильтров Блума (обычных) для множеств А и В (соответственно f_A и f_B), фильтром Блума для объединения А и В будет $f_A \vee f_B$, а фильтром пересечения будет $f_A \wedge f_B$.

При этом разность двух обратимых (но не устойчивых) фильтров Блума (count1 - count2, keyxor1 XOR keyxor2, valuesum1 - valuesum2) будет содержать разность множеств А и В со знаком + и разность множеств В и А со знаком -.

5 Текущие результаты

На данный момент (17.02.2022) изучена литература (Goodrich и Mitzenmacher [1]) по теме, а также на C++ реализован работающий обратимый фильтр Блума.

https://github.com/mmanchkin/bloom-lookup-table

(репозиторий приватный и нужно запросить доступ)

Эксперименты по оптимизации и сравнению можно будет начать, когда будет доступ к данным генома.

Список литературы

[1] Michael T. Goodrich и Michael Mitzenmacher. «Invertible Bloom Lookup Tables». В: *CoRR* abs/1101.2245 (2011). arXiv: 1101.2245. URL: http://arxiv.org/abs/1101.2245.