A1

1) В каких случаях алгоритмы работают очень долго или вообще могут не закончиться

Самый простой пример. Заполняем таблицу полностью, чтобы не осталось ни одной пустой ячейки NULL. Потом удаляем все элементы. После этого в таблице везде ERASED и нигде нет NULL. Если теперь искать ключ, которого нет, то цикл будет ходить по кругу и так и не встретит NULL

Ещё вариант, когда не бесконечно, но долго. Когда подряд много ERASED и занятых ячеек и только где то далеко есть NULL. Тогда поиск отсутствующего ключа будет проверять почти всё подряд

2) Как это исправить без перехеширования и больших переделок

Сделать ограничение по шагам. Например, максимум M проверок и если не нашли, то остановиться

В INSERT учитывать ERASED как свободное место. То есть запоминать первую ячейку ERASED и если дальше не нашлось ключа, то вставлять туда

Можно чуть улучшить SEARCH. Если по пути встретили ERASED, а потом нашли ключ, то можно ключ перенести в первую ERASED. Тогда в будущем будет меньше длинных цепочек

A2b

1) Как искать нейтральные строки длины 2

Идея: хэш для двух символов это что то вроде. Первый символ даёт одно число. Второй символ даёт другое число, и оно умножается на p

Нужно, чтобы суммарно получилось 0

Так как в формуле вычитается 'a', то для больших букв и цифр получается отрицательное значение. Это удобно, потому что можно подобрать один символ с минусом и второй с плюсом так, чтобы они взаимно уничтожились.

Как искать автоматически. Считаем значение v для всех разрешённых символов a..z, A..Z, 0..9. Сохраняем в словарь значение и символ. Дальше перебираем второй символ и считаем, какое значение нужно для первого. Если такое значение есть в словаре, то пара найдена

2) Нейтральные строки для p от 1 до 31

p 1 Zf

p 2 Zc

p 3 Zb

p 4 Xb

p 5 Vb

p 6 Za

p 7 Ya

p 8 Xa

p 9 Wa

p 10 Va

p 11 Ua

p 12 Ta

p 13 Sa

p 14 Ra

p 15 Qa

p 16 Pa

p 17 Oa

p 18 Na

p 19 Ma

p 20 La

p 21 Ka

p 22 Ja

p 23 Ia

p 24 Ha

p 25 Ga

p 26 Fa

p 27 Ea

p 28 Da

p 29 Ca

p 30 Ba

p 31 Aa

.

A3

1) Что я запускал и какие файлы получил

Я сделал генератор потока строк, точный подсчёт уникальных через unordered_set и оценку через HyperLogLog. Я запускал с параметрами B=10, streams=30, parts=20, N=50000. Программа вывела B=10, m=1024, theory_rel_1_04=0.0325 и theory_rel_1_30=0.040625. Также она сохранила файлы out_timeseries.csv и out_stats.csv

2) Какие графики строил

График 1 строится по out_timeseries.csv. По X там шаг или сколько элементов обработано. По Y там число уникальных. На графике две линии, true_unique и estimate

График 2 строится по out_stats.csv. По X там шаг или сколько элементов обработано. По Y там средняя оценка mean_est. Ещё добавляется полоса неопределённости, это mean_est плюс минус std_est

3) Комментарий по точности

В конце потока среднее истинное значение было 48092.2, а средняя оценка была 48227.6. Разница маленькая, примерно 0.28 процента

В среднем оценка чуть завышает. Среднее относительное смещение получилось около 0.36 процента. Максимально по шагам было около 1.17 процента, и минимум было около минус 0.30 процента

4) Комментарий по стабильности и теории

Относительный разброс можно смотреть как std_est делить на mean_est. В среднем получилось около 0.0336, а максимум был около 0.0372. Это выше 0.0325, но ниже 0.040625, то есть по смыслу попадает в ожидаемый диапазон

Если взять полосу mean_est плюс минус std_est, то mean_true попадал внутрь этой полосы на всех шагах. Это выглядит нормально, потому что оценка шумная, но не слишком

5) Почему выбрал B=10

B=10 это 1024 регистра. Это мало памяти, примерно около 1 килобайта под сами регистры. При этом точность получается примерно на уровне нескольких процентов, и это нормальный компромисс

6) Что можно улучшить

Самое простое улучшение это увеличить B. Тогда будет больше памяти, но меньше ошибка. Ещё можно увеличить streams, чтобы средние и отклонения считались стабильнее. Также можно поиграться с хэшем, но даже текущий вариант работает нормально

A4

1) Будет ли фильтр F(AB) давать положительный ответ для элементов из A пересечение B?

Да, будет. Если объект есть и в A, и в B, то при добавлении в оба фильтра он ставит одни и те же биты в 1. После побитового И эти биты останутся 1. Значит, проверка даст true

2) Будет ли F(AB) точно таким же, как фильтр, построенный только по A пересечение B?

Нет, не обязательно. Потому что в фильтре возможны коллизии по битам. Может случиться так, что элемент из A\B поставил какой то бит в 1, а элемент из B\A поставил тот же бит в 1. Тогда после И этот бит останется 1, хотя он не обязан быть 1 для пересечения

Простой пример. A это {x}. B это {y}. Пересечение пустое. Но если хэши x и y попали в одинаковые позиции, то F(A) и F(B) будут похожи. И после И получится не нулевой фильтр, хотя для пустого пересечения всё должно быть нули.