**Анализ алгоритма Бойера — Мура**

**Алгоритм поиска строки Бойера — Мура** — алгоритм общего назначения, предназначенный для поиска подстроки в строке. Разработан Робертом Бойером и Джеем Муром в 1977 году. Преимущество этого алгоритма в том, что ценой некоторого количества предварительных вычислений над шаблоном (но не над строкой, в которой ведётся поиск), шаблон сравнивается с исходным текстом не во всех позициях — часть проверок пропускаются как заведомо не дающие результата.

Общая оценка вычислительной сложности алгоритма Бойера — Мура – O(n + m), если не используется таблица стоп-символов, и если используется таблица стоп-символов – O(n + m + |E|), где n – длина строки, m – строка-образец, E – алфавит.

void BM(string X, string Y) { // Алгоритм Бойера — Мура

int i = 0;

vector <int> A;

if (Y.size() > X.size()) // Если строка образец больше строки

cout << "Строка-образец больше строки!" << endl;

else {

A = Prefix(Y);

while (i <= (X.size() - Y.size())) { // Пробегаемся по строке

int j = Y.size() - 1;

while (j >= 0 && Y[j] == X[i + j]) // Смотрим совпадения и смещяемся на символ назад

j--;

if (j < 0) { // Если нашлось, выводим номер

cout << i << " ";

if (i + Y.size() < X.size())

i += Y.size() - A[X[i + Y.size()]]; // И смещаем каретку для проверки

else // Иначе просто переходим к следующему символу

i++;

}

else // Иначе прибавляем единичку или разность номера + смещение

i += 1 > j - A[X[i + j]] ? 1 : j - A[X[i + j]];

}

}

}

В среднем сложность алгоритма равна http://students.uni-vologda.ac.ru/pages/pm15/zha/algoritm-boera-mura/image026.gif, но в худшем случае можно получить http://students.uni-vologda.ac.ru/pages/pm15/zha/algoritm-boera-mura/image028.gif. Для примера рассмотрим строки http://students.uni-vologda.ac.ru/pages/pm15/zha/algoritm-boera-mura/image030.gif и http://students.uni-vologda.ac.ru/pages/pm15/zha/algoritm-boera-mura/image032.gif. При таких входных данных каждую итерацию внешнего цикла i будет увеличиваться всего на единицу, а внутренний цикл будет выполняться http://students.uni-vologda.ac.ru/pages/pm15/zha/algoritm-boera-mura/image034.gif раз.

### **Достоинства**

Алгоритм Бойера — Мура на «хороших» данных очень быстр], а вероятность появления «плохих» данных крайне мала. Поэтому он оптимален в большинстве случаев, когда нет возможности провести предварительную обработку текста, в котором проводится поиск. Разве что на коротких текстах выигрыш не оправдает предварительных вычислений.

### **Недостатки**

Алгоритмы семейства Бойера — Мура не расширяются до приблизительного поиска, поиска любой строки из нескольких.

Сравнение не является «чёрным ящиком» (только если применяется эвристика стоп-символа), поэтому при реализации наиболее быстрого поиска приходится либо рассчитывать на удачную работу оптимизатора, либо вручную оптимизировать поиск на языке ассемблера.

Если текст изменяется редко, а поиск проводится часто (например, поисковой машиной), можно провести индексацию текста. Алгоритм поиска по индексу быстрееалгоритма Бойера — Мура.

На больших алфавитах (например, Юникод) таблица стоп-символов может занимать много памяти. В таких случаях либо обходятся хеш-таблицами, либо дробят алфавит, рассматривая, например, 4-байтовый символ как пару двухбайтовых, либо (что проще всего) пользуются вариантом алгоритма Бойера — Мура без эвристики стоп-символов.

Существует ряд модификаций алгоритма Бойера — Мура, нацеленных на ещё большее ускорение — например, турбо-алгоритм, обратный алгоритм Колусси и другие.