Анализ на задача “Кръстословица”

CodeIT, 2014-2015, Round2

Прочитайки задачата, по-опитните състезатели трябва да са забелязали, че е модификация на проблема за Multiple string matching. Формалната постановка на проблема е следната:

Даден е текст (с дължина N) и речник от M думи/шаблона (с обща дължина L). Искаме максимално ефективно да определим кои от думите се срещат в текста и къде.

Aho-Corasick е класически алгоритъм, който предоставя решение на тази задача. Подробен анализ може да бъде намерен на следните линкове:

<http://judge.openfmi.net:9080/mediawiki/index.php/Структури_за_низове>

<https://en.wikipedia.org/wiki/Aho-Corasick_string_matching_algorithm>

Алгоритъмът много напомня като идея друг класически алгоритъм за търсене на един(!) шаблон в текст, а именно алгоритъма на Knuth-Morris-Pratt. Общото между двата е, че се правят връзки (или линкове, т.нар. failure links), така че при добавяне на поредния символ от текста да можем да кажем кой е най-дългият префикс на дума от шаблона, който завършва на текущата ни позиция, до която сме стигнали. Тези два алгоритъма може на пръв поглед да изглеждат малко страшни, но можем да мислим за тях като един вид динамично оптимиране.

Ефективността на Aho-Corasick се крие зад помощната структура от данни Trie, която ползва за представяне на речника и добавянето на failure link-овете, за които стана въпрос по-рано.

Строенето на цялата структура, както и търсенето в нея, става за линейно време.

Единствената допълнителна стъпка, която трябва да се осъществи, за да завършим решението на конкретната задача “Кръстословица”, е да обходим матрицата която ни е дадена във всяка една от осемте посоки и да запазим на кои позиции и в кое направление сме срещнали всяка дума от речника.