Оптимизация памяти в алгоритме Смита — Уотермана

Никита Вяткин

20 марта 2024 г.

Пусть s_1 и s_2 — строки, локальное выравнивание которых мы ищем. А t_1 и t_2 — подстроки s_1 и s_2 соответственно, глобальное выравнивание которых есть локальное выравнивание s_1 и s_2.

Модифицируем алгоритм Смита — Уотермана следующим образом. При вычислении матрицы динамики будем хранить только две последних строчки. При этом, будем поддерживать максимальный вес выравнивания, когда-либо достигнутый, и позицию в матрице (j_1, j_2), где был такой вес. Таким образом, используя линейную память, мы найдём вес оптимального локального выравнивания и позиции правых концов t_1 и t_2 (j_1 и j_2 соответственно).

Теперь повторим всё то же самое для развёрнутых строк s_1 и s_2. Ясно, что вес оптимального локального выравнивания будет тот же, мы найдём позиции i_1 и i_2 левых концов t_1 и t_2.

Осталось найти глобальное выравнивание t_1 и t_2. Это можно сделать Алгоритмом Хиршберга, используя линейную память.

В итоге, мы модифицировали алгоритм Смита — Уотермана так, что он по прежнему работает за время $O(|s_1|\cdot|s_2|)$, но использует $O(min(|s_1|, |s_2|))$ дополнительной памяти.

Замечание: Позиции (i_1, i_2) начала подстрок t_1 и t_2 можно вычислять одновременно с (j_1, j_2), если вместе с предыдущей строкой матрицы динамики хранить координаты ячейки матрицы, начиная из которой можно прийти в данную с соответствующим score-ом выравнивания. Такой массив предков тоже можно пересчитывать динамически одновременно с матрицей динамики, и хранить мы тоже будем только две последние строки. Это избавляет от необходимости повторять алгоритм для развёрнутых строк, но на асимптотику по времени и памяти не влияет.