Тезисы к научно-исследовательской работе «Алгоритм нахождения наибольшей общей подпоследовательности для нескольких последовательностей Linear-MLCS»

Автор: Сидюк Дмитрий Андреевич, ученик 11 класса физико-математического профиля Ришельевского научного лицея г. Одесса.
Научный руководитель: Сидюк Андрей Анатольевич, Senior AI developer (math apparatus), ROMAD Cyber Systems Inc.

Цель

Целью данной работы является реализация алгоритма Linear-MLCS на языках программирования Python и C++, а так же исследование некоторых зависимостей в полученных с его помощью последовательностях.

Актуальность работы

Информация в различных своих проявлениях часто выражается в виде последовательностей элементов с конечным алфавитом, что порождает необходимость находить их наибольшие общие подпоследовательности. Будь то цепочка ДНК здорового человека и человека с генетическими отклонениями, паттерны поведения приложений и вредоносных программ, сравнение файлов и т.д. Для решения данной задачи было создано множество алгоритмов. Однако все эти решения объединяет серьезная проблема — они невероятно ресурсозатратные, требуют много памяти и времени для работы.

С современными темпами развития технологий объемы данных стремительно растут, что требует более оптимальных решений проблемы поиска наибольших общих подпоследовательностей. В данной работе будет рассмотрен алгоритм Linear-MLCS, который намного эффективнее аналогов в использовании места и времени необходимого для работы, к тому же он позволяет найти абсолютно все возможные наибольшие общие подпоследовательности *Multiple Longest Common Subsequence* (MLCS) для неограниченного количества последовательностей.

Ход работы

В работе описан принцип действия алгоритма Linear-MLCS на примере последовательностей S_1 = TFGACGADTC, S_2 = ATGLCTCAFG и S_3 = CTADGTALCG с практически выжным в биоинформатике алфавитом, состоящим из обозначений нуклеиновых оснований $\Sigma_4 = \{A, C, G, T\}$. Описание разбито на следующте этапы:

1. Избавление от уникальных элементов
2. Построение таблиц Successor Tables
3. Построение NCSG
4. Сортировка NCSG
5. Чтение NCSG

На каждом шаге обоснована необходимость данного шага, описан принцип действия, а так же предложена реализация на языках программирования Python и C++. Код программ выложен на GitHub и доступен по ссылке https://github.com/Garison1/MLCS-research.

Итоги исследования

За время выполнения работы мы реализовали алгоритм Linear-MLCS на языках программирования Python и C++, в том числе метод построения NCSG графа, его чтения, а так же методы прямой и обратной топологических сортировок. Описали работу алгоритма на примере его реализации на языке Python. Провели исследования зависимостей количества полученных LCS от их длины, определили распределение вероятностей длин LCS и их количества.

Список литературы

[1] Yanni Li и др. *A Real Linear and Parallel Multiple Longest Common Subsequences (MLCS) Algorithm*. 2016. URL: `https://www.kdd.org/kdd2016/papers/files/rpp0619-liA.pdf`.

Приложения

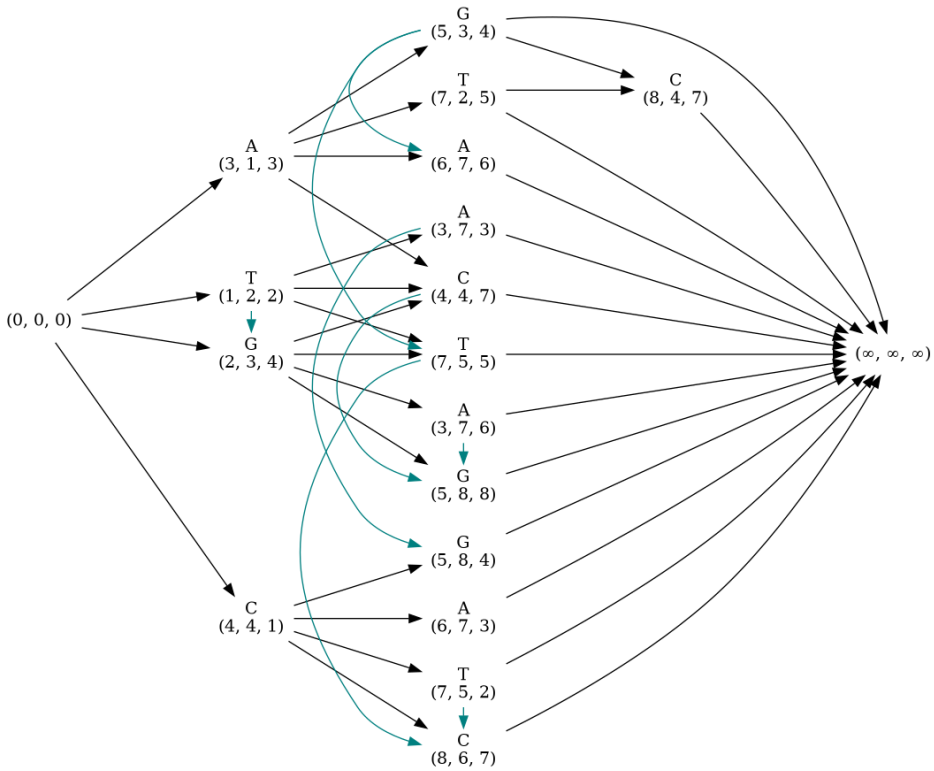
Приложение 1: Построенные Successor Tables для последовательностей S_1, S_2 и S_3

S_1	=	T	G	A	C	G	A	T	C
	0	1	2	3	4	5	6	7	8
A	3	3	3	6	6	6	-	-	-
C	4	4	4	4	8	8	8	8	-
G	2	2	5	5	5	-	-	-	-
T	1	7	7	7	7	7	7	-	-

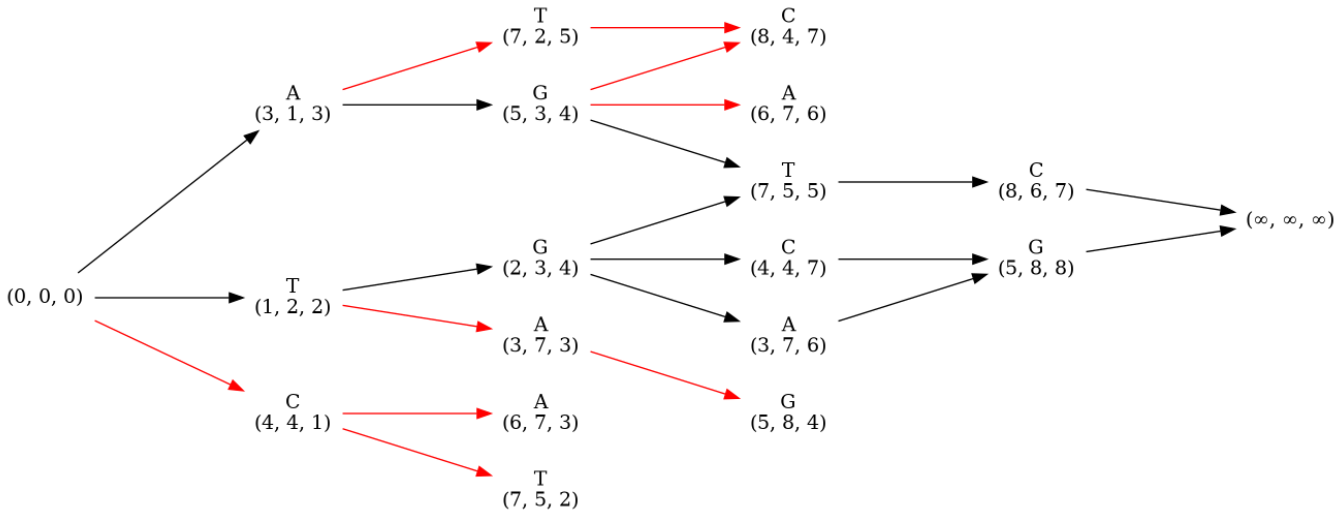
S_2	=	A	T	G	C	T	C	A	G
	0	1	2	3	4	5	6	7	8
A	1	7	7	7	7	7	7	-	-
C	4	4	4	4	6	6	-	-	-
G	3	3	3	8	8	8	8	8	-
T	2	2	5	5	5	-	-	-	-

S_3	=	C	T	A	G	T	A	C	G
	0	1	2	3	4	5	6	7	8
A	3	3	3	6	6	6	-	-	-
C	1	7	7	7	7	7	7	-	-
G	4	4	4	4	8	8	8	8	-
T	2	2	5	5	5	-	-	-	-

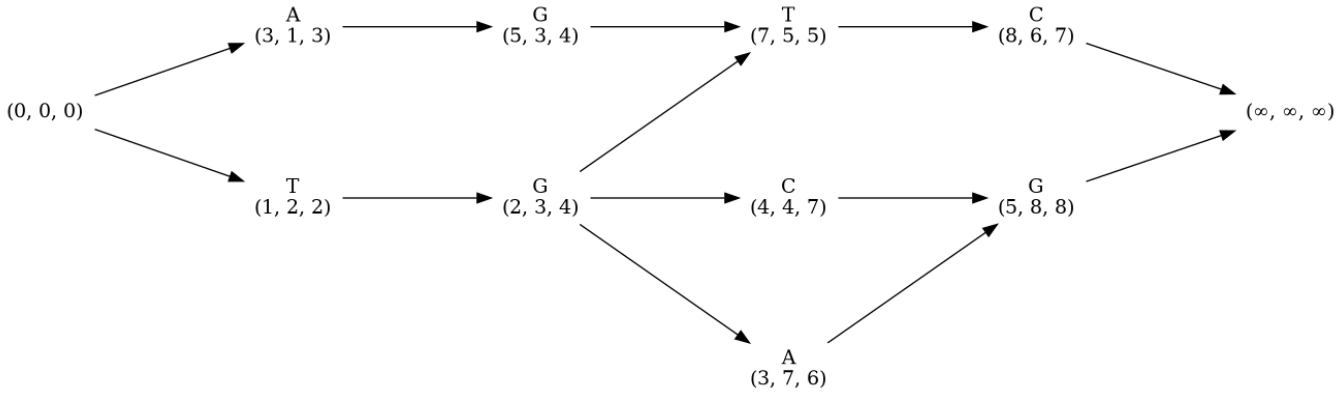
Приложение 2: Построенный *Non-redundant Common Subsequence Graph* (NCSG)



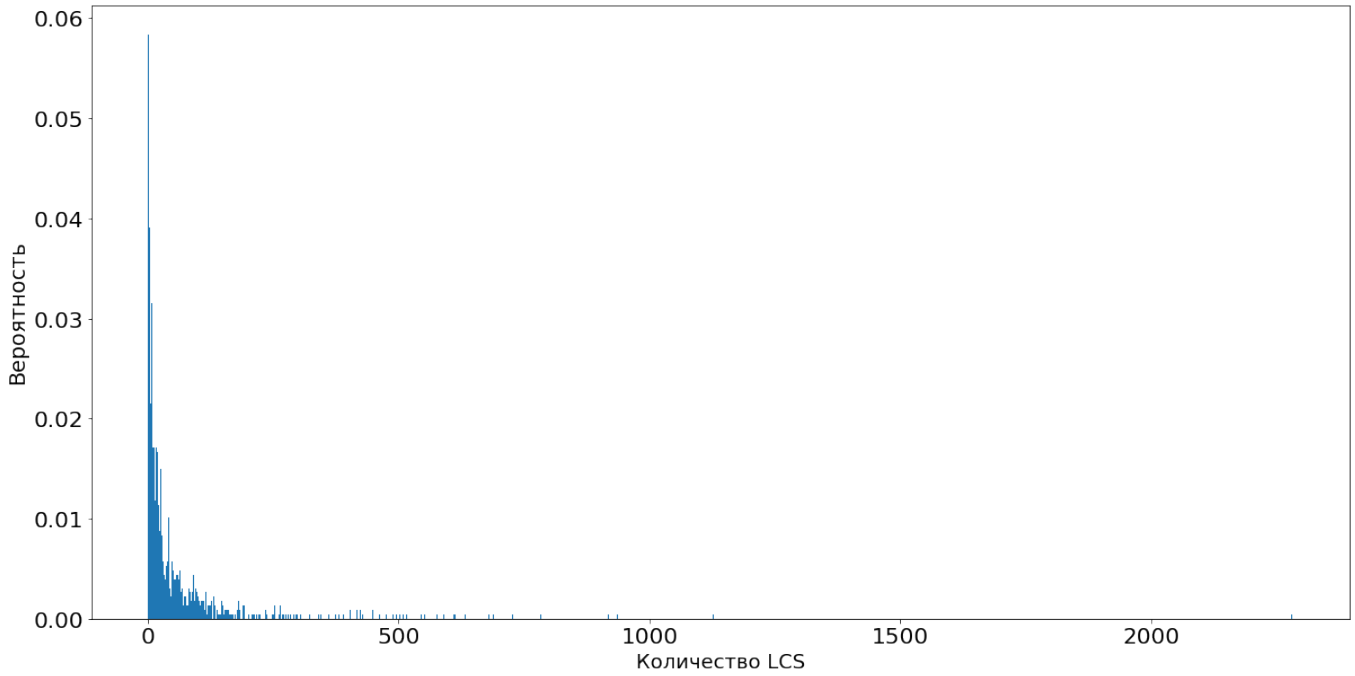
Приложение 3: NCSG отсортированный прямой топологической сортировкой



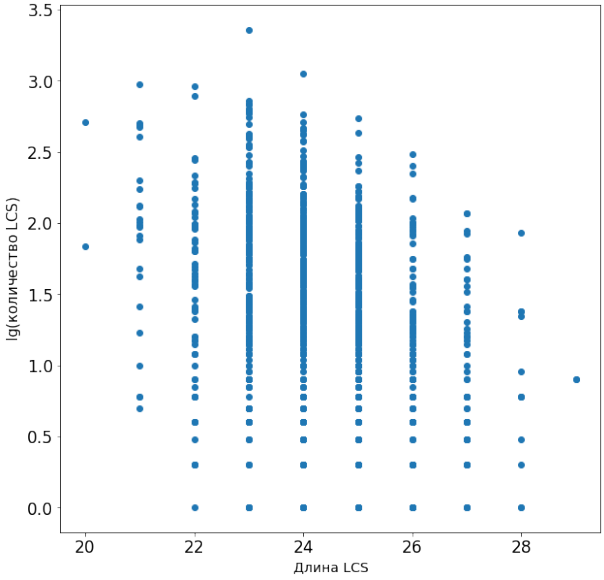
Приложение 4: NCSG отсортированный обратной топологической сортировкой



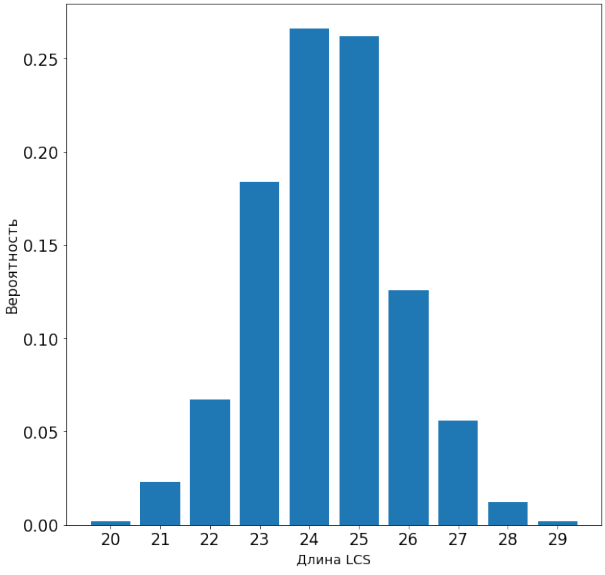
Приложение 5: Гистограмма вероятностей количества LCS



Приложение 6: Соотношение между длинами получаемых LCS и их количеством.



Приложение 7: Гистограмма распределения вероятностей длин LCS



Примечание: На приложениях 5, 6 и 7 изображены диограммы, полученные после анализа 1000 наборов из 3 последовательностей длиной 50 элементов и алфавитом 4.