Курсовая работа по курсу дискретного анализа: diff

Выполнил студент группы 08-308 МАИ *Попов Николай.*

1 Условие

Даны две строки со словами. Необходимо найти наибольшую общую подпоследовательность слов. Обратите внимание на ограничения по памяти, стандартное решение при помощи динамического программирования не подойдет.

2 Метод решения

Идея в основе этого алгоритма проста: если разделить входную последовательность $x = x_1x_2...x_m$ на две произвольные части по любому граничному индексу i на $x_b = x_1x_2...x_i$ и $xe = x_i + 1x_i + 2...x_m$, то найдется способ аналогичного разбиения y (найдется такой индекс j, разбивающий y на $y_b = y_1y_2...y_j$ и $y_e = y_j + 1y_j + 2...y_n$), такой, что $LCS(x, y) = LCS(x_b, y_b) + LCS(x_e, y_e)$.

Пока в последовательности x есть элементы, мы делим x пополам, находим подходящее разбиение для y и возвращаем сумму рекурсивных вызовов для пар последовательностей (x_b, y_b) и (x_e, y_e). Заметим, что в тривиальном случае, если x состоит из одного элемента и встречается в y мы просто возвращаем последовательность из этого единственного элемента x.

3 Описание программы

Программа реализует алгоритм нахождения наибольшей общей подпоследовательности (НОП) слов, используя алгоритм Хиршберга. Этот алгоритм является эффективным с точки зрения использования памяти методом для нахождения НОП, так как он требует линейного объема памяти относительно длины последовательностей.

3.1 Основные компоненты программы

- **Функция** `lcs_length`: Вычисляет длину НОП для двух векторов строк, которые представляют собой входные последовательности слов. Возвращает вектор целых чисел, где каждый элемент указывает максимальную длину НОП до данного индекса во второй последовательности.

- **Функция** `LCS_HIRSHBERG`: Реализует алгоритм Хиршберга для определения НОП между двумя последовательностями слов. Эта функция рекурсивно делит каждую последовательность на две части, вычисляет НОП для каждой пары частей, и комбинирует эти результаты.

- **Функция** `readWords`: Читает входные данные из стандартного ввода и преобразует их в вектор строк. Каждая строка интерпретируется как отдельное слово последовательности.
- **Главная функция** `main`: Контролирует выполнение программы, включая считывание входных данных, вызов функции `LCS_HIRSHBERG` для нахождения НОП и вывод результатов на стандартный вывод.

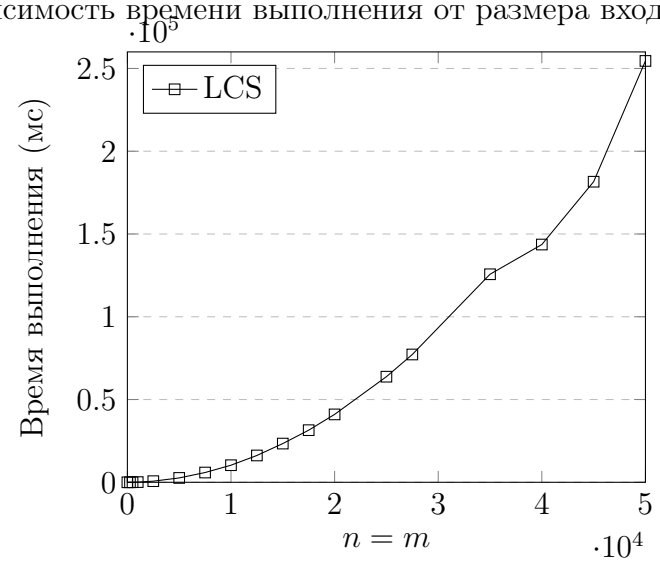
3.2 Принцип работы программы

1. Считываются две последовательности слов из стандартного ввода.
2. С использованием алгоритма Хиршберга вычисляется НОП.
3. Размер НОП и слова, составляющие НОП, выводятся на стандартный вывод.

Программа предназначена для демонстрации использования алгоритма Хиршберга в задаче нахождения НОП между двумя последовательностями слов. Она эффективно управляет памятью и подходит для работы с большими объемами данных.

4 Тест производительности

Зависимость времени выполнения от размера входных данных



Исходя из новых результатов тестирования, можно увидеть, что сложность описывается зависимостью $O(nm)$, где n, m — количество слов в тексте.

5 Выводы

В ходе разработки и тестирования программы, реализующей алгоритм Хиршберга для нахождения наибольшей общей подпоследовательности слов, были достигнуты следующие результаты:

1. Подтверждена высокая эффективность алгоритма Хиршберга в задачах, требующих оптимизации использования памяти. Реализация алгоритма показала, что он способен обрабатывать большие объемы данных, сохраняя при этом линейную сложность по памяти.

2. Разработка и тестирование программы обеспечили ценный опыт в реализации и оптимизации алгоритмов для обработки и анализа текстовых данных, подчеркнув важность выбора правильных алгоритмических и структурных решений.

Опыт, полученный в ходе работы над проектом, подчеркивает значимость эффективного управления ресурсами и оптимизации алгоритмов для работы с большими объемами данных. Реализация алгоритма поискка наибольшей общей подподследовательсти при помощи метода Хиршберга демонстрирует, как с помощью различных эвристик можно достигнуть значительного улучшения производительности и эффективности в вычислительных задачах.