Лабораторная работа № 7 по курсу: Дискретный анализ

Выполнил студент группы М8О-30ХБ-17 МАИ *Лопатин Александр.*

Задача

При помощи метода динамического программирования разработать алгоритм решения задачи, определяемой своим вариантом; оценить время выполнения алгоритма и объём затрачиваемой оперативной памяти. Перед выполнением задания необходимо обосновать применимость метода динамического программирования. Разработать программу на языке C или C++, реализующую построенный алгоритм. Формат входных и выходных данных описан в варианте задания.

Вариант 4: Игра с числом.

Имеется натуральное число n. За один ход с ним можно произвести следующие действия:

- Вычесть единицу;
- Разделить на два;
- Разделить на три.

При этом стоимость каждой операции - текущее значение n. Стоимость преобразования - суммарная стоимость всех операций в преобразовании. Вам необходимо с помощью последовательностей указанных операций преобразовать число n в единицу таким образом, чтобы стоимость преобразования была наименьшей. Делить можно только нацело.

Информация

Динамическое программирование – это способ решения сложных задач, путём разбиения их на подзадачи.Он применим к задачам с оптимальной подструктурой, выглядящим как набор перекрывающихся подзадач, сложность которых чуть меньше исходной. В этом случае время вычислений, по сравнению с "наивными" методами, можно значительно сократить.
Этапы построения алгоритма решения подзадач:

- Описать структуру оптимального решения.
- Составить рекурсивное решение для нахождения оптимального решения.

- Вычисление значения, соответствующего оптимальному решению, методом восходящего анализа.
- Непосредственное нахождение оптимального решения из полученной на предыдущих этапах информации.

Метод решения

Для того, чтобы решить данную задачу, можно пойти с другого конца: будем рекуррентно считать минимальные стоимости для каждого числа начиная с 1 и заканчивая n и записывать эти значения в массив dp. Будем считать, что значения dp[0] и dp[1] присвоены нулю. Тогда, для числа i следует рекуррентная формула:

а) если i кратно 6, то $i = i + min(dp[i-1], dp[i/2], dp[i/3])$

б) если i кратно 3, но не кратно 2, то $i = i + min(dp[i-1], dp[i/3])$

в) если i кратно 2, но не кратно 3, то $i = i + min(dp[i-1], dp[i/2])$

г) если i не кратно ни 2, ни 3, то $i = i + dp[i-1]$

Докажем справедливость этой формулы. Достичь числа i можно несколькими способами: либо если i кратно 3 перейти из числа i/3, либо если i кратно 2 перейти из числа i/2, либо перейти из числа i - 1. Очевидно, что оптимальным вариантом будет самый наименьший по стоимости путь, так как если выбрать не наименьший, то минимальной стоимость считаться уже не будет. Из этого следует вывод, что стоимости dp[i/3], dp[i/2] и dp[i - 1] также являются оптимальными (так как начальные значения dp[0] и dp[1] являются оптимальными).

Описание программы

Для вычисления значения минимальной стоимости числа n используется массив длиной n+1, который хранит значения оптимальной стоимости всех чисел от 0 до n, следовательно сложности по памяти и по времени составляют O(n).

- **lab7.cpp**
  Единственный файл, содержит в себе решение данной задачи.

Дневник отладки

Время	Описание
13.07 00:12:05	Было выведено наивное решение с помощью жадного алгоритма, которое в итоге не подходило к данной задаче
13.07 02:14:23	Описана рекуррентная формула
13.07 02:19:36	Программа исправно работает на нескольких тестах
13.07 02:28:57	Программа успешно прошла ряд тестов на производительность

Выводы

Динамическое программирование довольно распространено, в различных олимпиадах около половины задач решаются с помощью ДП. Всюду, где встречаются подзадачи, и где их можно легко выделить, динамическое программирование позволяет существенно ускорить работу программы. Метод является достаточно гибким, так как это не алгоритм, а метод построения алгоритмов.