Здравствуйте, меня зовут Закиров Вячеслав. Моя работа – исследования метода решения NP-полных задач.

Существует набор задач, которые пока решаются только полным перебором всех вариантов – такие задачи называются NP (в отличии от задач класса P, для которых известен полиномиальный алгоритм решения).

Среди них выделяются NP – полные задачи (, решение которых позволит получить алгоритм для всех остальных задач класса NP. Среди таких задач, я выбрал задачу о разбиении

Требуется разбить множество элементов на два подмножества с одинаковыми суммами. Как видно из рисунков алгоритм таких разбиений может быть существенно различным – например, в этом случае для успешного разбиения в каждую часть последовательно выбираются элементы от большего к меньшему. А в этой ситуации данный алгоритм не получит разбиения, но разбиение, как видно на рисунке, возможно, и для его получения используется другой алгоритм. Поэтому моей первой задачей было разработать универсальный алгоритм, который бы подходил для максимально возможного количества ситуаций.

В моем алгоритме я упорядочиваю начальное множество, а затем распределяю элементы от большего к меньшему на два множества. Если после размещения всех элементов суммы множеств не равны, я делаю шаг назад и меняю размещение предыдущего элемента (рекурсия). В своем исследовании я пытался определить границы эффективной применимости моего алгоритма и найти способы их расширить.

Базой для исследований я взял множества с количеством элементов до 800 и максимальном значением элемента до 10000 (этот диапазон был обусловлен техническими характеристиками моего компьютера). Полученные результаты работы алгоритма показали его хорошую эффективность на разделяемых множествах и бесполезность в случае неразделяемых множеств .

Это было вызвано тем, что алгоритм вынужден был проверять все варианты, как при полном переборе. При этом, он многократно проверял одни и те же ситуации, особенно, если количество элементов большое, а возможный диапазон их значений мал.

Для улучшения результатов работы алгоритма я ввел список «плохих ситуаций» - ситуация считается плохой, если она не приводит к разбиению. При повторном возникновении в процессе работы алгоритма «плохой ситуации» она не будет проверятся.

В результате этих изменений удалось увеличить эффективность работы алгоритма для неразделяемых множеств, а также немного улучшить результаты для разделяемых . Тем не менее, эффективность применения алгоритма по-прежнему сильно снижалась при переходе через «границу», когда максимальное значение элемента превышает количество элементов исходного множества.

Подводя итог, я могу сказать, что в результате моей работы был создан алгоритм, находящий разбиение эффективнее полного перебора на неразделяемых множествах и намного эффективнее – на разделяемых, при достаточно небольшом количестве элементов исходного множества (до 1000).

Благодарю за внимание.

ИССЛЕДОВАНИЯ МЕТОДА РЕШЕНИЯ NP-ПОЛНЫХ ЗАДАЧ

Существует набор задач, которые пока решаются только полным перебором всех вариантов - это NP задачи.

Задача коммивояжера
Кратчайшее решение «пятнашек» размера n*n
Раскраска графа
Задача о клике
Задачи о разбиении
Сапер, тетрис

Мой алгоритм: упорядочить начальное множество, а затем распределить числа от большего к меньшему на два массива. Если после размещения всех элементов суммы массивов не равны, я делаю шаг назад и меняю размещение предыдущего элемента (рекурсия).

Гипотеза: исследования заключалась в том, что данный алгоритм имеет ограниченную эффективность применения. В своем исследовании я хотел найти границы этой эффективной области и определить, возможно ли улучшить алгоритм для ее расширения.

Результаты исследований

Для разделяемых множеств

Алгоритм			
Максим. число			
10000	230	1862	1804
1000	248	659	850
100	143	108	801
10	18	101	108
Кол-во элементов	10	100	800

Полный перебор			
Максим. число			
10000	511	(2^99)-1	(2^799)-1
1000	511	(2^99)-1	(2^799)-1
100	511	(2^99)-1	(2^799)-1
10	511	(2^99)-1	(2^799)-1
Кол-во элементов	10	100	800

Таблица 1

Для неразделяемых множеств

Алгоритм			
Максим. число			
10000	511	2047	16383
1000	511	2047	16383
100	511	2047	16383
10	511	2047	16383
Кол-во элементов	10	12	15

Полный перебор			
Максим. число			
10000	511	2047	16383
1000	511	2047	16383
100	511	2047	16383
10	511	2047	16383
Кол-во элементов	10	12	15

Таблица 2

Как видно из таблицы №1, существенное увеличение количество шагов в работе алгоритма на разделяемых множествах происходит, когда максимальное значение чисел исходного множества превышает квадрат количества чисел во множестве. Для улучшения алгоритма было введено понятие «плохих ситуаций». Это повысило эффективность алгоритма, особенно для неразделяемых множеств

Для разделяемых множеств

Алгоритм			
Максим. число			
10000	230	1833	1693
1000	248	571	828
100	139	106	801
10	16	101	801
Кол-во элементов	10	100	800

Полный перебор			
Максим. число			
10000	511	(2^99)-1	(2^799)-1
1000	511	(2^99)-1	(2^799)-1
100	511	(2^99)-1	(2^799)-1
10	511	(2^99)-1	(2^799)-1
Кол-во элементов	10	100	800

Таблица 3

Для неразделяемых множеств

Алгоритм			
Максим. число			
10000	509	2046	16383
1000	509	2027	15363
100	487	1844	12534
10	217	785	2665
Кол-во элементов	10	12	15

Полный перебор			
Максим. число			
10000	511	2047	16383
1000	511	2047	16383
100	511	2047	16383
10	511	2047	16383
Кол-во элементов	10	12	15

Таблица 4

Вывод: в результате моего исследования был создан алгоритм, находящий разбиение произвольного множества эффективнее полного перебора на неразделяемых множествах и несравнимо эффективнее на разделяемых. При условии, что максимальное значение элемента множества не превышает количество чисел во множестве.
Результаты, полученные мною, могут быть использованы в задачах, где необходимо разделять сравнительно небольшое количество элементов (до 1000).