УДК 004.021 DOI: 10.17587/it.25.234-240

В. Е. Торчинский, доц., e-mail: vet@magtu.ru,
И. В. Торчинская, вед. инж.-программист, e-mail: asu@magtu.ru,
А. С. Файнштейн, канд. физ.-мат. наук, доц., e-mail: swetlana@mgn.ru,
С. И. Файнштейн, доц., e-mail: sfainshtein@yandex.ru,
Магнитогорский государственный технический университет им. Г. И. Носова

"Гибридный" алгоритм планирования государственных закупок товарно-материальных ценностей

Для планирования госзакупок товарно-материальных ценностей предлагается "гибридный" алгоритм набора заданной суммы из заданных стоимостей заявок (NP-полная задача "Сумма размеров"). Сумма набирается полиномиальным алгоритмом до тех пор, пока размер подзадачи не становится пригодным для решения точным алгоритмом. Такой "гибридный" алгоритм наследует вычислительную сложность и оценку абсолютной погрешности полиномиального алгоритма. В то же время применение на последнем этапе решения точного алгоритма позволяет набирать сумму с большой степенью точности. Данный алгоритм применяется при планировании госзакупок Магнитогорского государственного технического университета им. Г. И. Носова.

Ключевые слова: *NP-полнота, задача "Сумма размеров", приближенный алгоритм, "гибридный" алгоритм, планирование, государственные и муниципальные закупки*

Введение

Закупочная деятельность в нашей стране регламентируется двумя основными федеральными законами: 44-ФЗ [1] и 223-ФЗ [2]. С увеличением объема торгов существенно повышается трудоемкость работ по подготовке, проведению, учету и контролю проводимых торгов и заключаемых контрактов. Для повышения эффективности управления госзакупками необходима автоматизация планирования госзакупок с помощью современных информационных технологий.

Составление плана закупок товарно-материальных ценностей вуза начинается со сбора заявок, которые ежегодно подают все его подразделения. Далее заявки делятся на категории: канцтовары, оргтехника, мебель и др. Затем для каждой заявки определяется источник финансирования: федеральный бюджет, студенты-контрактники, программа стратегического развития и др. От этого зависит, под действие какого федерального закона попадает заявка и каким способом она должна быть реализована.

Чтобы сформировать план закупок, работнику контрактной службы необходимо сначала проанализировать поданные заявки и вручную проставить способы закупок для тех заявок, для которых важна закупка каким-то определенным способом. Этот момент можно пояснить на примерах.

Например, закупая канцтовары на аукционе, университет обязан купить канцтовары любой торговой марки у продавца, предложившего минимальную цену. Товары, чьи технические характеристики не важны, могут быть закуплены любым способом, в том числе и на аукционе. Для таких товаров изначально способ закупки не определен.

Если же технические характеристики товара важны (например, закупка серверов), то способ закупки — договор с единственным поставщиком. Заметим, что для таких товаров способ закупки проставляется вручную, и они не попадают в массив исходных данных, который формирует предлагаемый алгоритм.

Далее работник отдела выбирает способ закупки, сумму, которую необходимо набрать, и запускает алгоритм. Вначале алгоритм формирует рабочий массив заявок, из которых будет набираться требуемая сумма. Те заявки, у которых способ закупки не определен, и стоимость не превосходит набираемую сумму, автоматически входят в рабочий массив. Затем алгоритм выбирает из рабочего массива заявки, общая стоимость которых близка к заданной сумме.

Все заявки (слагаемые), участвующие в наборе суммы, равноправны и имеют единичные "веса", характеризующие степень их важности. Как уже было отмечено, наиболее важные закупки с единственным поставщиком изначально не попадают в массивы исходных данных, формируемых алгоритмом.

Для остальных товаров еще при подаче заявки указывается, в каком квартале подразделение желает получить товар. Эти сроки находятся в базе данных и учитываются алгоритмом при формировании исходных массивов. Товары, не вошедшие в набранную сумму, тоже будут закуплены в соответствующем квартале, но другим способом. Для них работник контрактной службы должен выбрать и вручную проставить другой способ закупки. После выбора способа закупки для таких товаров также возможно применение алгоритма.

Особенностью нашей задачи является ее сверхбольшая размерность: сумма в рублях имеет порядок $10^6...10^7$. В то же время число заявок сравнительно не велико, порядка нескольких сотен.

1. Математическая постановка задачи

Задача о наборе суммы

Условие. Заданы конечное множество A, целые положительные веса всех элементов $a \in A$ и положительные целые числа $Sum0$ и D.

Вопрос. Существует ли такое подмножество $A' \subseteq A$ такое, что суммарный вес его элементов Sum удовлетворяет условию $Sum0 - Sum \leqslant D$?

Заметим, что частным случаем этой задачи при $D = 0$ является модельная NP-полная задача "Сумма размеров" [3]. Таким образом, наша задача является классической NP-полной задачей. В оптимизационной постановке эту задачу можно сформулировать следующим образом: требуется набрать заданную целую положительную сумму из заданных целых положительных слагаемых так, чтобы не превысить заданную сумму и минимизировать отклонение набранной суммы от исходной.

2. Анализ существующих методов решения задачи о наборе суммы

Рассмотрим вначале точные методы решения данной задачи. Наиболее известным методом решения этой задачи является метод динамического программирования [4]. Известно, что с помощью этого метода задача "Сумма размеров" разрешима за псевдополиномиальное время [3]. Для решения задачи составляется таблица размера $n \times Sum$, где n — число элементов множества A, Sum — набираемая сумма, затем с помощью алгоритма вычислительной сложности $O(n \cdot Sum)$ набирается заданная сумма. Если Sum велико, работа алгоритма не завершится за реальное время.

Еще одним точным алгоритмом решения данной задачи является переборный алгоритм с возвратом [5], который в худшем случае является алгоритмом экспоненциальной вычислительной сложности и эффективно работает только при числе слагаемых $n \leqslant 20$. Так как рассматриваемая в данной работе задача может иметь сверхбольшую размерность, то для исходной задачи оба этих метода неприменимы.

Рассмотрим приближенные методы, позволяющие получать решение задачи "Сумма размеров" для большой размерности исходных данных. В первую очередь, это метод масштабирования для динамического программирования. Метод заключается в уменьшении всех заданных величин в $scale$ раз, где $scale$ — коэффициент масштабирования, и последующем округлении до целого путем отбрасывания дробной части. Погрешность полученного таким образом решения не превосходит $n \cdot scale$. Применим метод масштабирования к нашей задаче. Для того чтобы добиться существенного выигрыша в быстродействии, необходимо взять $scale$ порядка 1000, что даст недопустимо большую погрешность.

В работе [6] для отгрузки готовой продукции со склада листопрокатного цеха предлагается приближенный алгоритм набора сверхбольшой суммы с известной оценкой абсолютной погрешности. Но из-за специфики задачи в математической постановке есть дополнительные ограничения, кроме того, слагаемые определенным образом сгруппированы.

Таким образом, для задачи набора сверхбольшой суммы из заданных слагаемых с заданным допустимым отклонением возникает необходимость разработки нового эффективного приближенного алгоритма, а также оценка его абсолютной погрешности.

3. Описание "гибридного" алгоритма для задачи о наборе суммы

При первом запуске "гибридного" алгоритма исходный рабочий массив слагаемых упорядочивается по невозрастанию, при последующих запусках — перемешивается случайным образом. Подобные приемы "встряхивания" часто используются в эвристических алгоритмах локального поиска [7], которые имеют тенденцию застревать в точках локального оптимума. Число запусков ограничено константой, не зависящей ни от числа слагаемых, ни от набираемой суммы.

При наборе суммы, отклоняющейся от исходной суммы не более чем на *D*, алгоритм завершает работу. Полученное таким образом решение будем называть допустимым.

Без детализации работу алгоритма на одном запуске можно описать следующим образом. Если число слагаемых не более 20, то сумма набирается точным алгоритмом. Иначе запускается "жадный" FF-алгоритм (first fit, первый подходящий) [3]. Слагаемые берутся в том порядке, как они расположены в исходном массиве. Если слагаемое проходит ограничение по весу, то оно добавляется к текущему решению, иначе отбрасывается. Когда FF-алгоритм заканчивает работу, любое не вошедшее в решение "новое" слагаемое больше остаточной суммы (которую нужно добрать).

Затем оба массива (с набором текущей суммы и с неиспользованными "новыми" слагаемыми) сортируются в порядке невозрастания. Часть слагаемых в порядке неубывания удаляется из текущего решения и возвращается в рабочий массив. Слагаемые возвращаются в рабочий массив до тех пор, пока или остаточная сумма не станет больше либо равна минимальному "новому" слагаемому, или из текущего решения будут удалены все слагаемые. Затем последнее (максимальное) из перемещенных слагаемых безвозвратно удаляется из рабочего массива, после чего для набора остаточной суммы опять запускается либо точный алгоритм, либо FF-алгоритм.

Заметим, что удаление из рабочего массива на каждой итерации одного слагаемого, во-первых, гарантирует завершение одного запуска "гибридного" алгоритма не более чем за *n* итераций; во-вторых, позволяет получать новые наборы.

Ниже приведены основные шаги данного алгоритма.

Шаг 1. //новый запуск алгоритма на исходном массиве

Шаг 2. //вызов либо FF-алгоритма, либо точного алгоритма

Шаг 3. //FF-алгоритм

Шаг 4. //удаление слагаемых

Шаг 5. //формируем массив без удаленного слагаемого

Шаг 6. //точный алгоритм

4. Трассировка одного запуска алгоритма на модельном примере

Набираем сумму *Ves*0 = 100 с допустимым отклонением *D* = 0, рабочий массив *A* = [11(10), 8 (20), 3, 2(10)] (в скобках указано число слагаемых данного веса). *Ves* — текущая набранная сумма, *Sol* — текущий набор *Ves*, *Ost* — текущая остаточная сумма, *MinA* — минимальное слагаемое в массиве *A*.

1. FF-алгоритм (набирает сумму *Ost* = 100):
Ves = 99; *Ost* = 1; *Sol* = [11 (9)]; *A* = [11, 8 (20), 3, 2(10)]; *MinA* > *Ost*.

2. Возвращаем 11 из решения в массив *A*:
Ves = 88; *Ost* = 12; *Sol* = [**11 (8)**]; *A* = [11(2), 8 (20), 3, 2(10)]; *MinA* < *Ost*.

Удаляем 11 из массива *A*: *A* = [11(1), 8 (20), 3, 2(10)].

3. FF-алгоритм (набирает сумму *Ost* = 12):
Ves = 11; *Ost* = 1; *Sol* = [11]; *A* = [8 (20), 3, 2(10)]; *MinA* > Ost.

4. Возвращаем 11 из решения в массив *A*:
Ves = 0; *Ost* = 12; *Sol* = []; *A* = [11, 8 (20), 3, 2(10)]; *MinA* < *Ost*.

Удаляем 11 из массива *A*: *A* = [8 (20), 3, 2(10)].

5. FF-алгоритм (набирает сумму *Ost* = 12):
Ves = 11; *Ost* = 1; *Sol* = [8, 3]; *A* = [8 (19), 2(10)]; *MinA* > *Ost*.

6. Возвращаем 3 из решения в массив *A*:
Ves = 8; *Ost* = 4; *Sol* = [**8**]; *A* = [3, 2(10)] (в *A* входят только те слагаемые, которые не превосходят *Ost*); *MinA* < *Ost*.

Удаляем 3 из массива *A*: *A* = [2(10)], число слагаемых меньше 20.

7. Точный алгоритм (набирает сумму *Ost* = 4):
Ves = 4; *Ost* = 0; *Sol* = [**2, 2**];

8. *OptVes* = 100, *OptSol* = [**11(8), 8, 2, 2**].

5. Вычислительная сложность "гибридного" алгоритма

Точный алгоритм с возвратом используется для набора суммы только в том случае, если число слагаемых не превосходит 20. Время его работы не увеличивается с ростом числа слагаемых n или с увеличением исходной набираемой суммы, поэтому это время можно считать константой.

Число запусков "гибридного" алгоритма также ограничено константой и не зависит от размерности исходных данных, поэтому за вычислительную сложность примем число шагов "гибридного" алгоритма на одном запуске. Так как на каждой итерации одного запуска происходит удаление хотя бы одного слагаемого, число итераций не превосходит n.

Внутри одной итерации один раз вызывается FF-алгоритм с вычислительной сложностью $O(n)$, один раз сортируются два массива длины не более n, один раз происходит копирование массива длины не более n. Заметим, что в реальной задаче число слагаемых относительно невелико, поэтому сортировка за время $O(n^2)$ будет работать быстрее бинарной сортировки. Окончательно получаем вычислительную сложность $O(n^3)$.

6. Абсолютная и асимптотическая погрешности "гибридного" алгоритма

Будем называть решением полученный FF-алгоритмом набор элементов $A' \subseteq A$, а суммой Sum решения — сумму стоимостей (весов) входящих в него элементов. Обозначим $MaxTerm$ максимальный вес элемента $a \in A$ индивидуальной задачи I.

Определение. Назовем величину δ абсолютным отклонением набранной приближенным алгоритмом суммы от точного решения:

$$\delta = \begin{cases} Sum0 - D - Sum, & \text{если} \quad Sum < Sum0 - D; \\ 0, \quad \text{если} & Sum0 - D \leqslant Sum \leqslant Sum0. \end{cases}$$

Утверждение

1. Для всех индивидуальных задач I о наборе FF-алгоритмом заданной суммы $Sum0$ с заданным допустимым отклонением D и максимальным слагаемым $MaxTerm$ имеет место неравенство:

$$Sum0 - Sum \leqslant \delta, \text{ где } \delta \leqslant MaxTerm - D - 1.$$

2. Существует бесконечное число индивидуальных задач I, для которых $Sum0$ сколь угодно велико и $Sum = Sum0 - MaxTerm + D + 1$.

3. $R_{FF}^{\infty} = 1$, асимптотическая погрешность FF-алгоритма равна 1.

Доказательство пунктов 1—3 с помощью техники, описанной в работе [6], приведено в работе [8]. Таким образом, абсолютная погрешность "гибридного" алгоритма фиксирована и не увеличивается ни при возрастании числа слагаемых, ни при возрастании размера набираемой суммы.

7. Пример работы "гибридного" алгоритма на реальных данных

Для примера возьмем следующие входные данные:

1) период — 2016 г.;

2) категория товарно-материальных ценностей (ТМЦ) — строительные материалы;

3) тип финансирования — бюджет;

4) число заявок — 104.

Изначально все заявки имеют способ закупки "Не определен" (рис. 1). Общая стоимость всех заявок по категории — 2 002 986,4 руб.

Выставим сумму, которую требуется набрать (999 999 руб) и допустимое отклонение 100 руб. Алгоритм набирает сумму 999 951,6 руб (рис. 2), отклонение от заданной суммы равно 47,4 руб, что составляет 0, 0047 %.

Рассмотрим работу алгоритма более подробно. После запуска формируется рабочий массив A, в котором содержится информация о заявках — идентификационный номер ТМЦ и ее стоимость (вес). Для категории "Строительные материалы" было отобрано 125 заявок. Рассмотрим запуск алгоритма на данных, перемешанных случайным образом. Фрагмент массива приведен в табл. 1. Статус заявки "0" означает, что она не была включена в набор суммы.

Затем происходит набор суммы "жадным" алгоритмом. В результате набрана сумма 998 862 руб, в ее набор вошли 72 заявки. Остаточная сумма равна 1137 руб. Минимальное неиспользованное слагаемое $MinA$, которое осталось в исходном рабочем массиве, равно 1500.

Так как остаточная сумма превышает допустимое отклонение в 100 руб, начинаем возвращать в массив A слагаемые из решения до тех пор, пока остаточная сумма не превысит минимальное слагаемое $MinA$. Таким образом, элементы с весом 160, 168, 175 удаляются из набора и возвращаются в рабочий массив A, в табл. 2 им присвоен статус 1. После возврата остаток составляет 1640.

Затем создаем новый массив $NewA$, в который попадут элементы из A, не превышающие остаточную сумму, которую нужно набрать (1640). В массив попадут 8 элементов (табл. 2).

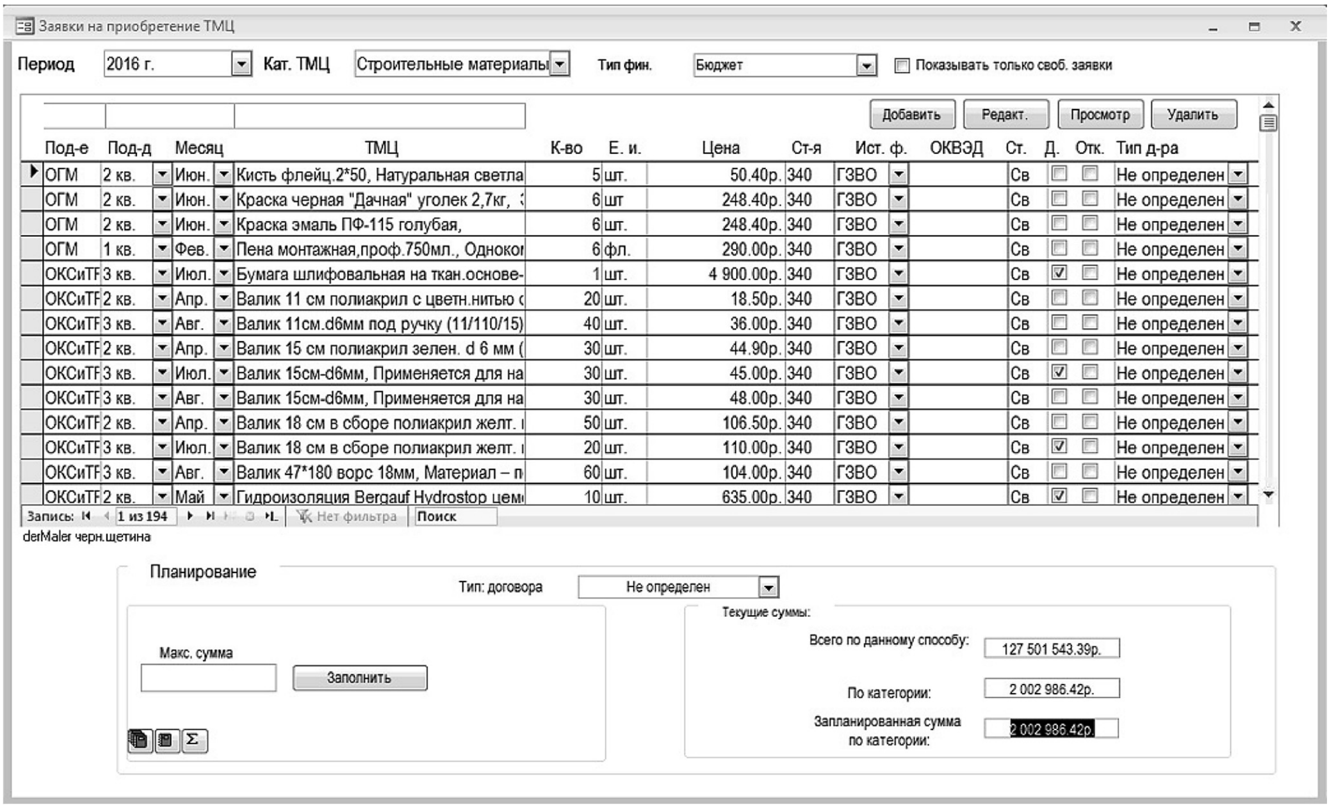

Рис. 1. Входные данные

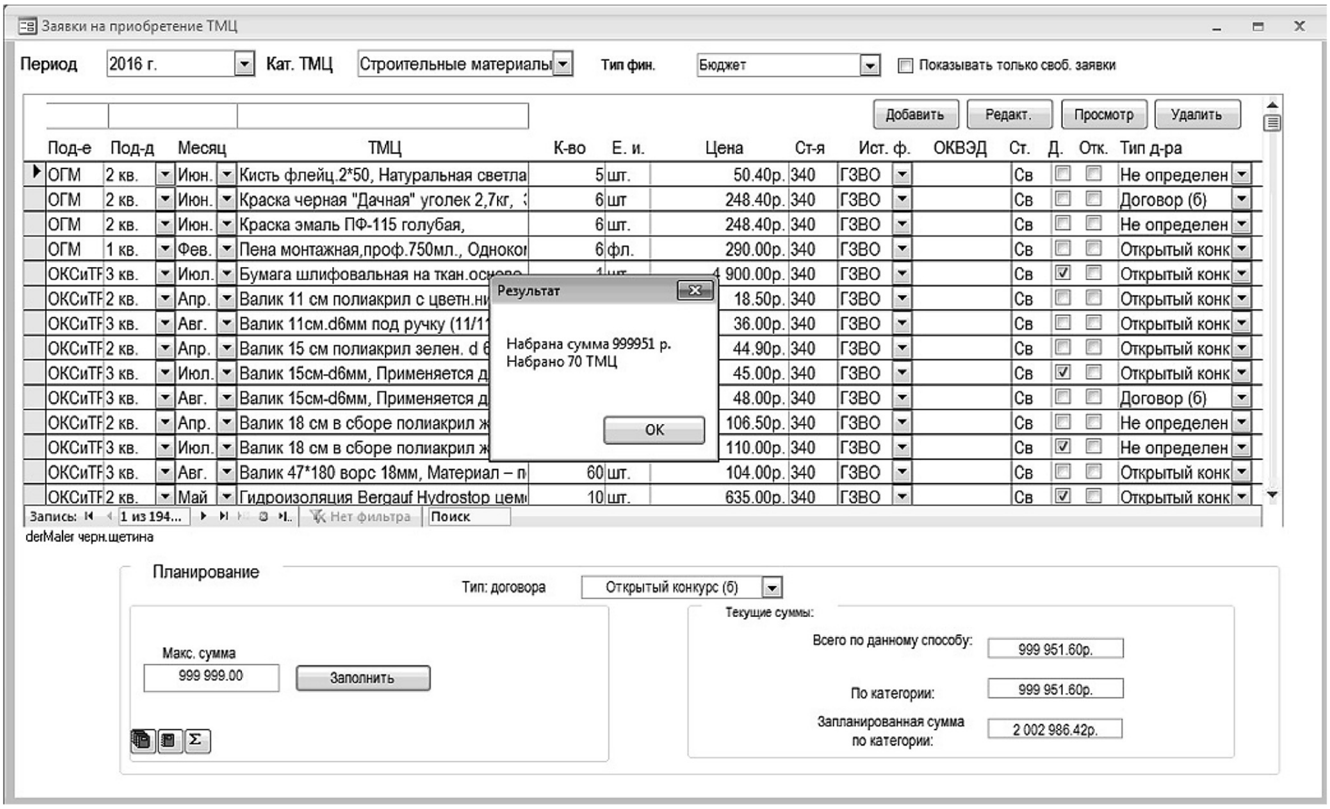

Рис. 2. Результаты работы программы

Таблица 1

Исходный массив *A*

Элемент	Статус	Номер ТМЦ	Индекс	Вес
A[1]	0	31 604	52	1680
A[2]	0	31 601	120	10 3510
A[3]	0	27 900	54	1890
A[4]	0	32 504	43	**1500**
A[5]	0	31 873	28	1200
...	...	...	...	...
A[125]	0	32 254	125	316 824

Таблица 2

Массив *NewA*

Элемент	Статус	Номер ТМЦ	Индекс	Вес
NewA[1]	1	31 743	1	160
NewA[2]	1	31 882	2	168
NewA[3]	1	27 886	3	175
NewA[4]	0	32 749	44	1517
NewA[5]	0	28 057	41	1500
NewA[6]	0	31 604	52	1680
NewA[7]	0	27 880	45	1530
NewA[8]	0	32 752	46	1592

Таблица 3

Результаты 10 запусков "гибридного" алгоритма

№	Сумма	Число элементов	Отклонение
1	938 028	118	61971
2	999 910	65	89
3	999 939	98	60
4	999 934	77	65
5	999 908	75	91
6	999 911	67	88
7	999 927	79	72
8	999 941	91	57
9	999 952	70	**47**
10	999 950	92	49

Так как число слагаемых не превышает 20, для набора остаточной суммы вызывается точный алгоритм. Точный алгоритм набирает сумму 1592 за счет одного элемента *NewA* [8]. Этот элемент добавляется к решению. Таким образом, набрана сумма 999 951 руб., в решение вошли 70 элементов (заявок).

Для получения различных решений и выбора наилучшего варианта запустим "гибридный" алгоритм 10 раз. Возвращаемая на каждом запуске набранная сумма, число слагаемых в наборе и отклонения приведены в табл. 3.

Как видно из табл. 3, на каждом запуске, кроме первого, алгоритм набрал сумму, удовлетворяющую заданному отклонению. Первый запуск алгоритма был проведен в массиве, отсортированном по невозрастанию, остальные — на перемешанных случайным образом.

Заключение

В "самом плохом случае" "гибридный" алгоритм набора заданной суммы наследует абсолютную погрешность "жадного" FF-алгоритма. Но благодаря добору остаточных сумм точным алгоритмом данный алгоритм хорошо работает на реальных данных. Предложенный алгоритм может применяться для решения задач малой, средней и большой размерности.

Список литературы

1. **Федеральный** закон от 05.04.2013 № 44-ФЗ "О контрактной системе в сфере закупок товаров, работ, услуг для обеспечения государственных и муниципальных нужд" // Собрание законодательства РФ. 08.04.2013. № 14. Ст. 1652.
2. **Федеральный** закон от 18.07.2011 № 223-ФЗ "О закупках товаров, работ, услуг отдельными видами юридических лиц" // Собрание законодательства РФ. 25.07.2011. № 30 (ч. 1). Ст. 4571.
3. **Garey M. R., Johnson D. S.** Computers and Intractability: A Guide to the Theory of NP-Completeness. San-Francisco: Freeman, 1979. 347 p. [**Гэри М., Джонсон Д.** Вычислительные машины и трудноразрешимые задачи. М.: Мир, 1982. 416 с.].
4. **Bellman R.** Dynamic Programming / Princeton, 1957.— 365 p. [**Беллман Р.** Динамическое программирование. М.: Иностранная литература, 1960. 400 с.].
5. **Lipski W.** Kombinatoryka dla programistów / 2 wyd. Warszawa: WNT, 1989. 188 p. [**Липский В.** Комбинаторика для программиста. М.: Мир, 1988. 200 с.].
6. **Девятов Д. Х., Файнштейн С. И., Тутарова В. Д., Калитаев А. Н.** Оперативное планирование отгрузки готовой продукции со складов металлургических предприятий // Мехатроника, автоматизация, управление. 2008. № 4. С. 36—40.
7. **Michalewicz Z., Fogel D.** How to Solve It: Modern Heuristics. Springer, Berlin, 2004. 554 p.
8. **Торчинская И. В., Файнштейн С. И.** Приближенный алгоритм набора заданной суммы с заданным отклонением // Межотраслевой институт Наука и образование. 2014. № 1. С. 84—86.

V. E. Torchinskiy, Assistant Professor, e-mail: vet@magtu.ru,
I. V. Torchinskaya, Leading Software Engineer of Department ACS of DIT&ACS., e-mail: asu@magtu.ru,
A. S. Fainshtein, Candidate of physical and mathematical Sciences, Assistant Professor, e-mail: swetlana@mgn.ru,
S. I. Fainshtein, Assistant Professor, e-mail: sfainshtein@yandex.ru,
Magnitogorsk state technical University named after G. I. Nosov

The "Hybrid" Algorithm of Planning Municipal Procurement of the Material Assets

In this work, for planning state procurement of goods-material values, the proposed "hybrid" algorithm sets the given amount of of the given values (NP-complete weighted set problem). The sum is collected by a fast "greedy" algorithm until the size of the subproblem becomes suitable for solving by an exact algorithm. This "hybrid" algorithm inherits the computational complexity and the absolute worst-case performance of the "greedy" algorithm. At the same time the use of the exact algorithm at the last stage of the solution allows to gain the sum with a high degree of precision. This algorithm is used in the planning of state procurement of MSTU named after G. I. Nosov.

Keywords: *NP-complete, weighted set problem, approximate algorithm, "hybrid" algorithm, planning, state and municipal procurements*

DOI: 10.17587/it.25.234-240

References

1. **Federal'nyj** zakon ot 18.07.2011 № 44-FZ *"O kontraktnoj sisteme v sfere zakupok tovarov, rabot, uslug dlja obespechenija gosudarstvennykh i munitsepal'nykh nuzhd"* (About contract system in the sphere of purchases of goods, works, services for ensuring the state and municipal needs), *Sobranie zakonodatel'stva RF*, 08.04.2013, no. 14, art. 1652. (in Russian).
2. **Federal'nyj** zakon ot 05.04.2013 № 223-FZ *"O zakupkakh tovarov, rabot, uslug otdel'nymi vidami juridicheskikh lits"* (About purchases of goods, works, services by separate types of legal entities), *Sobranie zakonodatel'stva RF*, 25.07.2011, no. 30 (part 1), art. 4571. (in Russian).
3. **Garey M. R., Johnson D. S.** Computers and Intractability: A Guide to the Theory of NP-Completeness, San-Francisco, Freeman, 1979, 347 p.
4. **Bellman R.** Dynamic Programming, Princeton, 1957, 365 p.
5. **Lipski W.** Kombinatoryka dla programistów, 2 wyd., Warszawa, WNT, 1989, 188 p. (in Russian).
6. **Devjatov D. Kh., Fainshtein S. I., Tutarova V. D., Kalitaev A. N.** Operativnoe planirovanie otgruzki gotovoj produktsii so skladov metallurgicheskikh predprijatij (Operational planning of shipment of finished products from warehouses of metallurgical enterprises), *Mekhatronika, Avtomatizatsiya, Upravlenie*, 2008, no. 4, pp. 36—40 (in Russian).
7. **Michalewicz Z., Fogel D.** How to Solve It: Modern Heuristics, Springer, Berlin, 2004, 554 p.
8. **Torchinskaya I. V., Fainshtein S. I.** Absolute and asymptotic performance for the approximation algorithm finding predetermined sum with a predetermined deviation, *Mezhotraslevoj institut Nauka i obrazovanie*, 2014, no. 1, pp. 84—86 (in Russian).