

sii v usloviyakh razvitiya informatsionno-kommunikatsionnykh tekhnologiy : materialy nauchno-prakticheskoy konferentsii [Innovations in the Conditions of Development of Information and Communication Technologies. Proceedings of the Scientific and Practical Conference], Moscow, Moscow State Institute of Electronics and Mathematics, National Research University Higher School of Economics Publ. House, 2009, pp. 229–232.

9. Uvaysov S. U., Yurkov N. K. Metodika obespecheniya teplovoy kontroleprigodnosti radiotekhnicheskikh ustroystv na etape proyektirovaniya [Technique of providing a thermal testability of radio engineering devices at design stage]. *Vestnik Samarskogo gosudarstvennogo tekhnicheskogo universiteta. Seriya: Fiziko-matematicheskiye nauki (Rossiyskaya Federatsiya)* [Bulletin of the Samara State Technical University. Series: Physical and Mathematical Sciences (Russian Federation)], 2012, no. 7, pp. 16–22.

10. Uvaysov S. U., Kofanov Yu. N., Manokhin A. I. *Modelirovaniye teplovykh protsessov pri proyektirovanii, ispytaniyakh i kontrole kachestva radioelektronnykh sredstv* [Modeling of thermal processes at design, tests and quality control of radio-electronic means], Moscow, Moscow State Institute of Electronics and Mathematics, National Research University Higher School of Economics Publ. House, 1998. 139 p.

11. Uvaysov S. U., Segen A. V., Kofanov Y. N., Pyatnitskaya G. A. *Primeneniye kompyuternogo izmeritelnogo teplovizora KRIT_T i matematicheskogo modelirovaniya dlya obespecheniya nadezhnosti i kachestva radioelektronnykh sredstv* [Use of computer measuring thermal imager KRIT_T and mathematical modeling for ensuring reliability and quality of radio-electronic means], Moscow, Moscow State Institute of Electronics and Mathematics, National Research University Higher School of Economics Publ. House, 1997. 121 p.

12. Khatskevich O. P., Ivanov I. A., Uvaysov S. U. Innovatsionnyy podkhod k sintezu testov dlya diagnostirovaniya analogovykh skhem [Innovative approach to synthesis of tests for diagnosing of analog schemes]. *Innovatsionnyye tekhnologii, nauchnyye i tekhnicheskiye dostizheniya, ikh pravovaya zashchita : sbornik statey IV Mezhdunarodnoy nauchno-prakticheskoy konferentsii (g. Tolyatti, 26–28 maya 2011 g.)* [Innovative Technologies, Scientific and Technical Achievements, Their Legal Protection. Proceedings of the IV International Scientific and Practical Conference (Tolyatti, 26–28 May 2011)], Tolyatti, Moscow, 2011, pp. 298–302.

13. Shalumov A. S., Manokhin A. I., Shalumova N. A. *Modelirovaniye teplovykh protsessov v tekhnicheskikh obektakh s pomoshchyu avtomatizirovannoy podsistemy ASONIKA-T* [Modeling of thermal processes in technical objects by means of the automated ASONIKA-T subsystem], Kovrov, Kovrov State Technological Academy named Degtyarev Publ. House, 2002. 140 p.

14. FloTHERM. Perform thermal analysis, create virtual models, and test design modifications of electronic equipment before physical prototyping. *Mentor Graphics*. Available at: <http://www.mentor.com/products/mechanical/flotherm/flotherm/>.

15. HyperLynx Thermal. Fast, accurate 3D modeling and simulation of thermal impact of PCB placement and routing. *Mentor Graphics*. Available at: <http://www.mentor.com/pcb/hyperlynx/thermal/>.

УДК: 519.863

ПРИМЕНЕНИЕ МЕТОДОВ ЛИНЕЙНОЙ ОПТИМИЗАЦИИ ДЛЯ ВЫРАВНИВАНИЯ ОСТАТКОВ ДЕНЕЖНЫХ СРЕДСТВ НА ЛИЦЕВЫХ СЧЕТАХ В ПРОЦЕССЕ ОСУЩЕСТВЛЕНИЯ НАЧИСЛЕНИЯ ПЛАТЫ ЗА ЖКУ

Статья поступила в редакцию 13.12. 2014, в окончательном варианте 04.02. 2015

Федосин Александр Сергеевич, аспирант, Мордовский государственный университет, 430005, Российская Федерация, г. Саранск, ул. Б. Хмельницкого, 39, nsdxfela@gmail.com

В статье представлен метод решения проблемы издержек расщепления платежей, возникающей при осуществлении начисления платы за ЖКУ в крупных биллинговых системах. Приведено подробное описание сути проблемы, а также последствия, к которым она может привести на практике. Предлагаемый автором метод основан на использовании математических инструментов линейного

программирования. Рассмотрена постановка задачи оптимизации. Описаны детали и технические аспекты реализации метода для биллинговой системы, применяемой в ООО «Саранский Расчетный Центр». Приведено общее описание использованного программного обеспечения, а также инструмента моделирования. Проанализирован ряд вопросов применения математической библиотеки Microsoft Solution Framework совместно с СУБД Microsoft SQL Server, лежащей в основе подсистемы хранения данных упомянутой биллинговой системы. Статья содержит результаты использования разработанного решения для выравнивания остатков денежных средств на лицевых счетах жильцов из заданного множества многоквартирных жилых домов за конкретный расчетный период.

Ключевые слова: линейная оптимизация, модель, симплекс-метод, ЖКХ, биллинг, взаимозачет, клиринг, выравнивание остатков, издержки расщепления, OML

APPLICATION OF LINEAR OPTIMIZATION FOR OFFSETTING OF MONEY BALANCES IN ACCOUNTS DURING UTILITY BILLING PROCESS

Fedosin Aleksandr S., post-graduate student, Mordovia State University, 39 B. Khmelnit-skiy St., Saransk, 430005, Russian Federation, e-mail: nsdfxela@gmail.com

The article describes an approach for solving payments “distribution costs” problem that takes place in process of utility billing for citizens, with using of large billing systems. The detailed description of problem and some of its hypothetical consequences is presented. Described method is based on linear programming techniques usage. Much attention is given to the formulation of optimization problem. The technical details of proposed method implementation designed for billing system of LLC "Saransk Clearing Centre" are provided. The general descriptions of used software and modelling tools are noted. Attention is drawn to the questions of using the Microsoft Solution Framework with RDBMS Microsoft SQL Server which is the basement of said billing's storage subsystem. Data are given about results of built solution applying for money balances offsetting in accounts of given set of apartment houses in given settlement period.

Keywords: linear optimization, model, simplex-method, utility services, billing, netting, clearing, offsetting of money balances, distribution costs, OML

Задачи линейного программирования нашли широкое применение в различных сферах экономики и техники. Возможности программных библиотек, реализующих численные методы решения этих задач, делают их привлекательными для использования при обработке больших массивов данных в крупных информационных системах. Целью данной работы является разработка методики применения инструментов линейного программирования для решения задачи о взаимозачетах в рамках биллинговой системы ООО «Саранский расчетный центр».

Общая характеристика проблематики статьи. Проблемы, связанные со сбытом услуг населению являются общеизвестными для всех поставщиков ЖКУ. Результатом неудачной организации этого процесса зачастую становится возникновение крупной дебиторской задолженности [6]. В такой ситуации привлекательной для поставщиков становится возможность пользоваться услугами так называемых «расчетно-кассовых центров», или РКЦ. В отличие от чисто платежных систем основные функции расчетных центров составляют полный биллинговый цикл:

- ведение баз данных, необходимых для выставления счетов на оплату;
- обработка показаний приборов учета (счетчиков);
- начисления по тарифам и показаниям счетчиков с учетом скидок, недопоставок, перерасчетов;
- печать и доставка единых платежных документов (ЕПД) потребителям услуг;
- взаимодействие с платежными системами;

- диспетчеризация и распределение платежей;
- подготовка отчетов по начислениям и сборам;
- претензионная работа с плательщиками [5].

Зачастую оплата населением услуг при использовании единых платежных документов подразумевает необходимость распределения поступающих денежных сумм между поставщиками (в отличие от схемы, при которой собранными денежными средствами распоряжается управляющая компания). В идеальном случае для конкретного лицевого счета ситуация должна выглядеть следующим образом: если существует задолженность за квартплату, то это означает, что задолженность есть по каждой из статей (например, горячему водоснабжению, газу, электроснабжению и т.д.). Однако на самом деле в процессе ведения баз данных и выставления счетов на оплату неизбежна ситуация, при которой по тому или иному лицевому счету необходимо произвести перерасчет за предыдущие периоды. Причем перерасчеты могут производиться по отдельным услугам (например, за горячую воду, если выяснилось, что в определенный момент ее качество не соответствовало нормам или ее подача прекращалась на период больше допустимого). В результате по ряду статей образуются переплаты, тогда как по другим статьям может иметься задолженность. При этом сальдо (совокупный остаток по статьям) на лицевом счете отдельного абонента может быть нулевым, положительным (задолженность), отрицательным (переплата). Потребитель, не знакомый с принципами расщепления оплаты, но в то же время регулярно вносящий нужные суммы, указанные в графе «итого» платежных документов, может быть озадачен тем фактом, что у него имеется задолженность перед одной из ресурсоснабжающих организаций. Подобная ситуация в рассматриваемой предметной области иногда называется «издержками расщепления» [2].

Для организации, отвечающей за биллинг, вышеописанная ситуация является нежелательной еще и по причине того, что отчеты, которые формируются на основе имеющихся данных, могут представлять искаженную картину структуры задолженностей. В данной работе предлагается в качестве решения алгоритм, в основе которого лежит симплекс-метод. Алгоритм обеспечивает построение моделей с использованием выражений на языке OML, основываясь на данных биллинга. Используя решатель SimplexSolver (входящий в состав фреймворка Solver Foundation), алгоритм определяет значения переменных, соответствующие оптимальному плану. Затем происходит процесс записи операций в базу данных. Ниже описывается вариант реализации алгоритма, разработанный для автоматизированной системы начислений ООО «Саранский Расчетный Центр».

Методика выполнения расчетов. Основными ограничениями, в рамках которых должен действовать алгоритм, являются следующие:

1. Общее сальдо по лицевому счету в результате не должно изменяться (очевидно, что сумма «к оплате» в квитанции должна оставаться прежней).
2. Не должно изменяться общее сальдо по поставщику в рамках одного дома (например, если дом имеет некоторую сумму задолженности перед поставщиком, предоставляющим коммунальные услуги горячего водоснабжения и отопления, то эта сумма после работы алгоритма должна остаться неизменной).
3. Вываживание остатков выполняется на множестве лицевых счетов абонентов по различным статьям, но всегда в пределах одного конкретного многоквартирного дома.

По итогам проведенных операций на статьях лицевых счетов суммы переплаты и задолженности уменьшаются, что не влияет на размер сальдо по лицевым счетам.

В качестве примера ситуации, требующей использования процедуры «расщепления» можно привести следующий случай: потребитель №1, исправно (своевременно) вносящий

плату за ЖКУ, видит в своем платежном документе, что на его лицевом счете имеется задолженность за электроэнергию в размере 300 руб., и в то же время переплата по статье «Центральное отопление» на сумму 350 руб. (ситуация проиллюстрирована в таблице). Сальдо по лицевому счету в этом случае составляет – 50 руб. Жилец того же дома, потребитель №2 имеет переплату по статье «Электроснабжение» на сумму 300 руб. и задолженность по статье «Центральное отопление» на сумму 350 руб. Сумма задолженностей перед поставщиком электроэнергии составляет 300 руб., сумма переплат 300 руб., сальдо по поставщику (если принять, что в доме всего две квартиры) – 0 руб. Для поставщика услуги «Центральное отопление» эти суммы соответственно равны 350, 350 и 0 руб.

Таблица

Данные, иллюстрирующие ситуацию с возникновением «издержек расщепления»

		Электроснабжение		Центральное отопление	
Потребитель №1		300 руб.		–350 руб.	
Потребитель №2		–300 руб.		350 руб.	
Сумма переплат	Сумма задолженностей	–300 руб.	300 руб.	–350 руб.	350 руб.
Сальдо по поставщику		0 руб.		0 руб.	

Мы, очевидно, можем взаимно «компенсировать» задолженности потребителя при условии сохранения сальдо по поставщику и лицевым счетам. Иными словами, уменьшая задолженность потребителя №1 по электроснабжению на сумму переплаты потребителя №2 по этой же услуге, а также «гася» задолженность потребителя №2 по отоплению за счет переплаты потребителя №1, мы получим, что суммы задолженностей и переплат по обеим статьям теперь равны нулю, а сальдо по поставщику и по лицевым счетам остались неизменными.

Приведенный пример, конечно же, не может отражать масштабы реальных задач. Рассмотрим, например, 500-квартирный дом. Услуги этому дому предоставляет 5–7 поставщиков, а суммы задолженностей и переплат варьируются от копеек до десятков тысяч рублей по одной квартире (абоненту). При таких показателях для решения задачи выравнивания сальдо целесообразно прибегнуть к инструментам линейного программирования, определив сумму задолженностей по услугам лицевых счетов как линейный функционал, который нам предстоит минимизировать при заданных линейных ограничениях.

Следует также упомянуть, что в литературе по тематике линейной оптимизации можно встретить отдельную разновидность задач, именуемых «задачами о клиринге» [4]. Клиринг – это процедура зачета взаимной задолженности между предприятиями. Цель решения такой задачи представляет собой максимизацию доходов от взаимозачета. По мнению автора, модель задачи о клиринге достаточно близка по своей сути задаче, описываемой в данной работе. Однако, с точки зрения предметной области, процесс оптимизации издержек расщепления называть клирингом в точном смысле этого слова нельзя.

Постановка задачи оптимизации. Рассмотрим общую задачу линейного программирования [1]:

$$f(x) = c^{(1)}x^{(1)} + c^{(2)}x^{(2)} + \dots + c^{(n)}x^{(n)}$$

при условиях:

$$\left. \begin{aligned} x^k &\geq 0, k \in I_+, \\ a_{11}x^{(1)} + a_{12}x^{(2)} + \dots + a_{1n}x^{(n)} &\leq b^{(1)}, \\ \dots \\ a_{m1}x^{(1)} + a_{m2}x^{(2)} + \dots + a_{mn}x^{(n)} &\leq b^{(m)}, \\ a_{m+1,1}x^{(1)} + a_{m+1,2}x^{(2)} + \dots + a_{m+1,n}x^{(n)} &= b^{(m+1)}, \\ \dots \\ a_{s1}x^{(1)} + a_{s2}x^{(2)} + \dots + a_{sn}x^{(n)} &= b^{(s)}, \end{aligned} \right\} \quad (1)$$

где $c^{(j)}, b^{(i)}, a_{ij}, i = \overline{1, s}, j = \overline{1, n}$ – заданные числа, I_+ – заданное подмножество индексов из множества $\{1, 2, \dots, n\}$.

Цель работы описываемого алгоритма – минимизация сумм задолженностей по услугам лицевых счетов абонентов в рамках многоквартирного дома. Исходными данными в контексте этой задачи являются суммы задолженностей и переплат, сгруппированные по лицевым счетам и поставщикам. То есть для каждого лицевого счета данные будут представлены $n \times 2$ переменными, где n – количество уникальных поставщиков услуг, предоставляемых потребителям. Например, если по статье «Горячее водоснабжение» мы имеем задолженность в размере 100 руб., а по статье «Отопление» – переплату на сумму 200 руб. (обе услуги предоставляются одним поставщиком), то при решении оптимизационной задачи это будет выражено двумя переменными, $x_1 = 100$ р. и $x_2 = -200$ р. Обе переменные в дальнейшем могут только уменьшаться по модулю.

Отметим также еще одно немаловажное обстоятельство: все переменные в нашей задаче отражают величины, измеряемые в денежных единицах (рублях). В базах данных для хранения подобных значений чаще всего прибегают либо к специальным «денежным», либо к обычным вещественным типам. В биллинговой системе, на базе которой разрабатывалось решение, в таком случае используется тип данных `double`, позволяющий хранить 64-разрядные значения в форме с плавающей запятой. Вместе с тем ситуация, в которой значения имеют более двух знаков после запятой, приводит нас в затруднение: мы не имеем возможности работать с суммами (значениями переменных) менее одной копейки. Выйти из этого положения можно с помощью следующих действий:

1. На этапе формулирования задачи в аналитическом виде все суммы представить с целыми числами копеек (по сути, умножив на 100 изначальные значения, выраженные в рублях с копейками).

2. Добавить к условиям задачи еще одно – условие целочисленности для «денежных» переменных.

3. Полученные результаты преобразовать к вещественным типам данных, разделив на 100.

Теперь введем следующие обозначения.

Пусть $c_n^{(k)}x_{cn}^{(k)}$ – сумма задолженности, образовавшаяся на лицевом счете жилого помещения (абонента) n перед поставщиком k , а $d_n^{(k)}x_{dn}^{(k)}$, в свою очередь, – сумма переплаты, где $c_n^{(k)}$ и $d_n^{(k)}$ – заданные числа.

В таком случае суммарное сальдо для лицевого счета n перед поставщиком k мы можем представить в форме разности $(d_n^{(k)}x_{dn}^{(k)} - c_n^{(k)}x_{cn}^{(k)})$.

Используя введенные обозначения, определим вид функционала, который нам предстоит *минимизировать*:

$$\sum_{i=1}^n \sum_{j=1}^k c_i^{(j)} x_{ci}^{(j)} = f(x),$$

$$x_{cn}^{(k)} \geq 0. \quad (2)$$

Кратко это можно записать так:

$$f(x) \rightarrow \inf, x \in X. \quad (3)$$

Следует заметить, что в качестве альтернативы минимизации суммы задолженностей мы вполне можем рассматривать *максимизацию* линейной функции, представляющей собой сумму переплат:

$$\sum_{i=1}^n \sum_{j=1}^k -d_i^{(j)} x_{di}^{(j)} = f(x),$$

$$x_{dn}^{(k)} \geq 0. \quad (4)$$

Или же, по аналогии с (3):

$$f(x) \rightarrow \sup, x \in X.$$

Ограничения диктуются следующими правилами:

1. Сальдо (совокупный остаток) по каждому лицевому счету должно остаться неизменным:

$$\sum_{j=1}^k (c_i^{(j)} x_{ci}^{(j)} - d_i^{(j)} x_{di}^{(j)}) = S_{ai},$$

$$x_{ci}^{(j)} \geq 0, x_{di}^{(j)} \geq 0,$$

$$i = \overline{1, n}, x_{ci}^{(j)}, x_{di}^{(j)} - \text{целые} \quad (5)$$

где S_{ai} – заданное целое число.

2. Сальдо по поставщику в рамках многоквартирного дома также должно остаться неизменным:

$$\sum_{i=1}^n (c_i^{(j)} x_{ci}^{(j)} - d_i^{(j)} x_{di}^{(j)}) = S_p^{(j)},$$

$$x_{ci}^{(j)} \geq 0, x_{di}^{(j)} \geq 0, j = \overline{1, k},$$

$$x_{ci}^{(j)}, x_{di}^{(j)} - \text{целые} \quad (6)$$

где $S_p^{(j)}$ – заданное целое число.

3. Выше было отмечено, что в процессе выравнивания сальдо остатки по отдельным статьям лицевых счетов должны или оставаться неизменными, или уменьшаться по модулю. Иными словами, если мы имеем переплату по отоплению в размере –200 руб., то в результате оптимизации мы должны получить на выходе остаток по данной статье в интервале $[-200; 0]$ руб. Это приводит к следующим ограничениям:

$$x_{ci}^{(j)} \leq s_{ci}^{(j)},$$

$$x_{di}^{(j)} \leq s_{di}^{(j)},$$

$$i = \overline{1, n}, j = \overline{1, k},$$

$$x_{ci}^{(j)}, x_{di}^{(j)} - \text{целые} \quad (7)$$

где $s_{ci}^{(j)}$ и $s_{di}^{(j)}$ – суммы остатков по статьям лицевого счета i , на которых имеются соответственно задолженности и переплаты относительно услуг поставщика j .

Ограничения, подобные (5) и (6) в литературе принято называть ограничениями типа равенств, (7) – ограничениями типа неравенств [3].

Результатом работы алгоритмов оптимизации должны стать значения переменных $x_{cn}^{(k)}$, при которых сумма модулей задолженностей будет минимальна. Однако, с точки зрения биллинговой системы, это не дает окончательного решения нашей проблемы. Для того, чтобы его достигнуть, необходимо, чтобы остатки денежных средств по различным статьям услуг лицевых счетов стали равны найденным значениям. Иными словами, конечной операцией станет изменения баланса по статьям услуг поставщика k на лицевом счете n на суммы, равные соответственно $\hat{x}_{cn}^{(k)} - x_{cn}^{(k)}$, где $\hat{x}_{cn}^{(k)}$ – соответствующий баланс до проведения процедуры оптимизации.

Реализация алгоритма. Решение описанной выше проблемы было реализовано и испытано авторами статьи для автоматизированной системы начислений платы за ЖКУ, используемой в ООО «Саранский расчетный центр». Далее приведено общее описание для ряда технических вопросов, которые могут представлять интерес в контексте решения задачи выравнивания остатков.

Программное обеспечение, для которого разработан алгоритм, построено с использованием технологии .NET. При этом в качестве основы подсистемы хранения данных применяется широко известная реляционная СУБД Microsoft SQL Server. Эти обстоятельства во многом определили выбор математической библиотеки, использованной при разработке программы Microsoft Solver Foundation (MSF), являющейся мощным инструментом построения оптимизационных моделей и поддерживающей полноценный .NET API. Среди важных преимуществ данной библиотеки следует отметить встроенный декларативный язык OML, специально созданный для задач моделирования [9].

При использовании встроенных алгоритмов симплекс-метода в MSF целевая функция и ограничения решаемой задачи могут быть переданы в соответствующий класс (Model) в виде строк, содержащих описанные при помощи правил OML параметры модели. Это позволяет без лишних затрат ресурсов генерировать нужные выражения непосредственно на стороне БД. Хотя с точки зрения принципов разработки программного обеспечения подобный выбор может показаться нестандартным (СУБД обычно мало приспособлены для задач конкатенации множества строк). Но практика показывает, что производительность приложения в целом не страдает на реальных объемах данных. Более того, мы имеем преимущество, заключающееся в отсутствии необходимости передачи больших объемов информации между подсистемой хранения данных и приложением (которые появились бы в случае, если ограничения для модели строились не на стороне БД). Вся логика извлечения и конструирования параметров для модели размещена в хранимых процедурах БД биллинга. Причем параметрами процедур являются уникальный идентификатор дома и период, относительно которого мы производим операции выравнивания.

Следует отметить, что данные в рассматриваемой биллинговой системе (как и во всех реальных информационных системах) в результате целого ряда обстоятельств не всегда бывают корректными. Поэтому перед решением любых задач, затрагивающих тем или иным образом большие объемы данных, требуется оценить качество этих данных и по возможности выполнить операции, максимально его повышающие или же каким-либо образом сократить масштаб негативного влияния некорректных сведений на конечный результат [8].

Далее рассмотрим процесс построения выражений, описывающих целевую функцию, параметры и ограничения.

Самый первый шаг – описание параметров (переменных). На этом этапе мы просто формируем список переменных длиной $n \times m \times 2$, где n – число лицевых счетов, m – число поставщиков услуг. Для удобства каждая переменная будет иметь в своем имени литеру “с” (credit) или “d” (debt) и соответствовать сумме задолженности или переплаты. Кроме того, с помощью агрегатных функций языка TSQL рассчитываем суммы остатков на лицевых счетах по статьям различных поставщиков – чтобы в итоге получить все необходимые нам ограничения типа неравенств [7].

Упрощенный запрос, составленный в псевдокоде, приведен в листинге на рисунке 1 (имена атрибутов для удобства выделены жирным шрифтом).

```
select
  number
, rsoCode
, 'x' + typeDC +
  cast( ROW_NUMBER () over (order by number desc) as varchar(40)) as variable
, 'x' + typeDC + cast( ROW_NUMBER () over (order by number desc) as varchar(40)) +
  case when typeDC = 'c' then ' <= '
  else ' >= ' end + cast (saldo *100 as varchar(150)) as eq1
, 'x' + typeDC + cast( ROW_NUMBER () over (order by number desc) as varchar(40)) +
  case when typeDC = 'c' then ' >= 0'
  else ' <= 0' end as eq2
into #result
from vars
```

Рис. 1. Листинг запроса, формирующего описание параметров и получающего суммы остатков на лицевых счетах (записан в псевдокоде)

Результат работы этого запроса представлен в виде таблицы, на рисунке 2.

	number	code	variable	eq1	eq2
142	2220114720	19	xd142	xd142 >= -90273.00	xd142 <= 0
143	2220114720	19	xc143	xc143 <= 20427.00	xc143 >= 0
144	2220114720	7	xc144	xc144 <= 13266.00	xc144 >= 0
145	2220114720	3	xc145	xc145 <= 12359.00	xc145 >= 0

Рис. 2. Пример результата выполнения запроса из листинга рисунка 1

Такой запрос вернет таблицу, содержащую среди прочих три столбца типа «VARCHAR»: сам параметр (variable); верхнее (eq1) и нижнее (eq2) ограничения (в соответствии с указанными ограничениями по формуле (7)). Здесь typeDC – тип значения, хранимого в переменной (задолженность/переплата), number – номер лицевого счета, saldo – сумма остатка. Предполагается, что табличное выражение vars в листинге 1 содержит подзапрос, возвращающий перечисленные значения и скрывающий все нюансы бизнес-логики. Умножение сумм на сто объясняется тем, что мы решаем целочисленную задачу, обеспечивая тем самым исключение варианта получения дробного числа копеек.

Отдельные процедуры отвечают за формирование на языке OML ограничений (5) и (6). В этих случаях с помощью несложных запросов соответственно считаются суммы по поставщикам и лицевым счетам. Одновременно в аналитическом виде строятся неравенства. Причем левая часть неравенства по сути представляет собой все переменные, сумму значений которых мы рассчитали с помощью агрегатной функции SUM и группировки по поставщикам или лицевым сче-

там. Отсутствие в языке TSQL агрегатной функции, предназначенной для конкатенации строк, потребовало поиска альтернативного решения, которым стала конструкция FOR XML.

Приведем пример запроса, который построит необходимые нам ограничения (6). Напомним, что их смысл сводится к необходимости сохранять неизменным сальдо по поставщику услуги в ходе минимизации задолженностей:

```
select code, crd, cast(f.Content as varchar(max)) + ' == ' + cast(crd as varchar(max)) as eq
from (
    select code, sum(saldocrd) as crd
    from #t t
    group by code
) i
cross apply
(
    SELECT ' + ' + w.variable
    FROM   #t AS w
    WHERE  w.code = i.code
    FOR XML PATH(''),
    TYPE
) as f(Content)
```

Рис. 3. Листинг запроса, формирующего ограничения типа (6) (записан в псевдокоде)

	code	crd	eq
1	3	1286731.00	+xc5+xc9+xc13+xc18+xc23+xd28+xc32+xc36 == 1286731.00
2	7	940037.00	+xc4+xc8+xc12+xc17+xd21+xc22+xc27+xc31+xc35 == 940037.00
3	19	-811647.00	+xc3+xd7+xd11+xd16+xd20+xd26+xd30+xd34 == -811647.00
4	29	432015.00	+xd1+xc2+xd6+xc10+xd14+xc15+xc19+xd24+xc25+xc29+xc33 == 432015.00

Рис. 4. Пример результата запроса по листингу, соответствующему рисунку 3

Этот запрос использует данные, сформированные с помощью кода, представленного в листинге 1. Причем предполагается, что данные хранятся во временной таблице #t. Мы используем суммы для получения общего сальдо по поставщику (табличное выражение *i* в листинге 2) и имена переменных для построения уравнения (табличное выражение *f* в листинге 2). Аналогичным образом строится запрос для получения ограничений типа равенств (5).

Результаты. Среди наиболее очевидных и простых метрик, с помощью которых можно оценить эффективность предложенного решения, удобно выбрать абсолютную сумму $\Delta x_{ci}^{(j)}$, на которую уменьшилась задолженность в конкретном доме, а также соответствующую ей относительную величину:

$$\delta_x = (\Delta x_{ci}^{(j)} / x_{ci}^{(j)}) \times 100\%. \quad (8)$$

Описанный метод и программное обеспечение применялись для оптимизации издержек расщепления относительно 1976 многоквартирных домов (МКД), начисления платы за ЖКУ для которых производятся в ООО «Саранский Расчетный Центр». Процедура выполнялась на данных системы биллинга, актуальных на август 2014 г.

Время выполнения расчета (проведения оптимизации) составило приблизительно 6 ч. с учетом следующих технических характеристик компьютера: процессор Intel Core i5-4570 с тактовой частотой 3,2 GHz, объем ОЗУ – 16 Гб.

Сокращение общей суммы задолженности по множеству МКД составило 10,3 % относительно изначальной суммы (при сохранении общего сальдо). Гистограммы, отражающие результаты оптимизации более детально, представлены на рисунках 5 и 6.

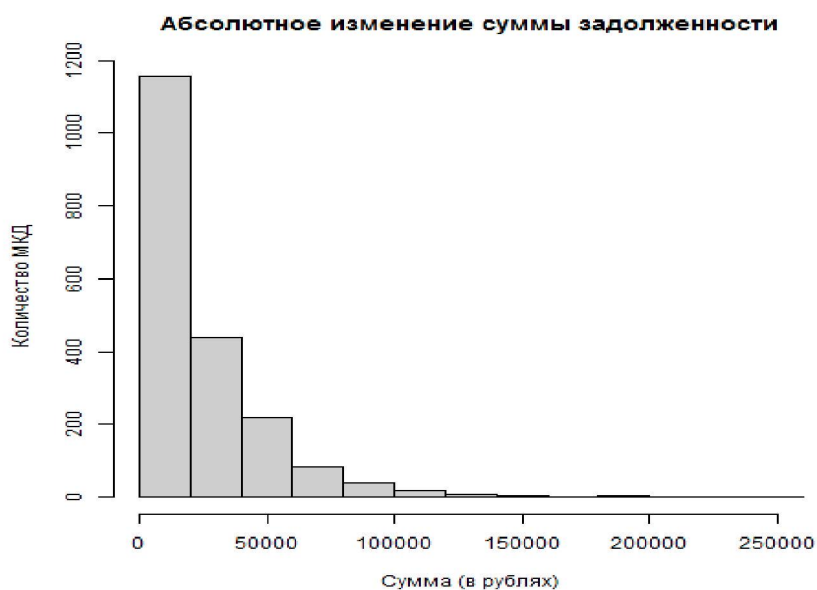


Рис. 5. Абсолютное изменение суммы задолженности в результате проведения оптимизации для указанного множества домов

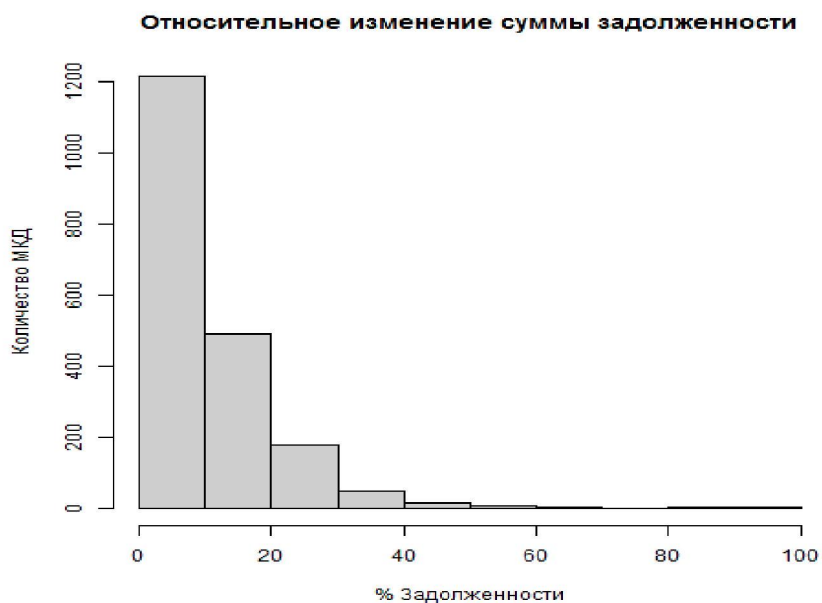


Рис. 6. Относительное изменение суммы задолженности в результате проведения оптимизации для указанного множества домов

Следует отметить, что с точки зрения задач ведения учета и выставления счетов населению вышеупомянутое множество домов не отличается единообразием: дома управляются различными компаниями (причем нередко эти компании сменяют друг друга во времени); однотипные ресурсы могут предоставляться различными поставщиками, структура задолженности и ее дальнейшая судьба зачастую регулируется целой совокупностью факторов юридического и административного характера. В этом, во многом, и проявляются хорошо известные и характерные для отрасли в целом проблемы. Однако тот факт, что, несмотря на указанные трудности, применение метода позволило значительно улучшить ситуацию, говорит о жизнеспособности предложенной модели оптимизации. Кроме того, нельзя забывать, что оптимизация в данном случае выполнялась только для одного расчетного периода (август 2014 г.), и её результаты позволяют с уверенностью говорить о целесообразности выполнения этой процедуры регулярно, непосредственно перед выставлением счетов на оплату населению.

Список литературы

1. Ашманов С. А. Линейное программирование / А. С. Ашманов. – Москва : Наука, 1981. – 340 с.
2. Васильев Ф. П. Линейное программирование / Ф. П. Васильев, А. Ю. Иваницкий. – Москва : Факториал, 1998. – 176 с.
3. Бурносова И. Коварная переплата: разбираемся в квитанции на оплату ЖКХ / И. Бурносова // Аргументы и факты. – Режим доступа: <http://www.irk.aif.ru/zkh/gorod/81571>, свободный. – Заглавие с экрана. – Яз. рус. (дата обращения 20.11.2014).
4. Лапитова А. Т. Применение линейного программирования в исследовании социально-экономических процессов / А. Т. Лапитова. – Челябинск : Южно-Уральский государственный университет, 2010. – 123 с.
5. Лисина Т. Г. Единые информационные расчетные центры как инфраструктурное обеспечение реформы жилищно-коммунального хозяйства / Т. Г. Лисина // CustIS : заказные инфорсистемы. – Режим доступа: <http://lib.custis.ru/Articles-lisina-iovrf11-2007.htm>, свободный. – Заглавие с экрана. – Яз. рус. (дата обращения 20.11.2014).
6. Рахтеенко В. Е. Какой расчетный сервис нужен поставщикам ЖКУ / Рахтеенко В. Е. // ЖКХ: Журнал руководителя и главного бухгалтера. – 2005. – № 11, ч. 1. – С. 21–24.
7. Селко Д. SQL для профессионалов. Программирование / Д. Селко. – Москва : Лори, 2009. – 442 с.
8. Федосин А. С. Проблемы качества данных в автоматизированных системах коммерческого учета потребления энергоресурсов / А. С. Федосин, А. В. Савкина // Прикаспийский журнал: управление и высокие технологии. – 2014. – № 2. – С. 158–164.
9. Solver Foundation / DevLabs. – Available at: <http://msdn.microsoft.com/en-us/devlabs/hh145003.aspx> (accessed 20.11.2014).

References

1. Ashmanov S. A. *Lineynoe programmirovaniye* [Linear Programming], Mosow, Nauka Publ., 1981. 340 p.
2. Vasilev F. P. *Lineynoe programmirovaniye* [Linear Programming], Mosow, Factorial Publ., 1998. 176 p.
3. Kovarnaya pereplata: razbiraemsya v kvitantsii na oplatu ZhKKh [Insidious overpayment: investigation utility bill]. *Argumenty i fakty* [Arguments and Facts]. Available at: <http://www.irk.aif.ru/zkh/gorod/81571> (accessed 20 November 2014).
4. Lapitova A. T. *Primeneniye lineynogo programmirovaniya v issledovanii sotsialno-ekonomicheskikh protsessov* [Applications of linear programming in socioeconomic processes research], Chelyabinsk, South Ural State University Publ. House, 2010. 123 p.
5. Lisina T. G. *Edinye informatsionnye raschetnye tsentry kak infrastrukturnoe obespechenie reformy zhilishchno-kommunalnogo hozyaystva* [Unified Information settlement centers as infrastructure support of reform of housing and communal services]. *CustIS: the custom information systems*. Available at: <http://lib.custis.ru/Articles-lisina-iovrf11-2007.htm> (accessed 20 November 2014).

6. Rakhtenko V. Ye. Kakoy raschetnyy servis nuzhen postavshchikam ZhKU [What kind of billing service do utility services providers need]. *ZhKKh: Zhurnal rukovoditelya i glavnogo bukhgaltera* [Housing and Communal Services: Magazine of Director and Chief Accountant], 2005, no. 11, part 1, pp. 21–24.
7. Selko D. *SQL dlya professionalov. Programmirovaniye* [SQL for smarties. Programming]. Moscow, Lori Publ., 2009. 442 p.
8. Fedosin A. S., Savkina A. V. Problemy kachestva dannykh v avtomatizirovannykh sistemakh kommercheskogo ucheta potrebleniya energoresursov [Data quality issues for energu management systems]. *Prikaspiyskiy zhurnal: upravlenie i vysokie tekhnologii* [Caspian Journal: Management and High Technologies], 2014, no. 2, pp. 158–164.
9. Solver Foundation. *DevLabs*. Available at: <http://msdn.microsoft.com/en-us/devlabs/hh145003.aspx> (accessed 20 November 2014).

УДК 004.94, 519.872

ИМИТАЦИОННЫЕ МОДЕЛИ КЛАСТЕРОВ И ГРИД-СИСТЕМ¹

Статья поступила в редакцию 10.12. 2014, в окончательном варианте 08.02. 2015

Гаевой Сергей Владимирович, аспирант, Волгоградский государственный технический университет, 400005, Российская Федерация, г. Волгоград, пр. Ленина, 28, e-mail: gaevserge@mail.ru

Фоменков Сергей Алексеевич, доктор технических наук, профессор, Волгоградский государственный технический университет, 400005, Российская Федерация, г. Волгоград, пр. Ленина, 28, e-mail: saf@vstu.ru

Лукьянов Виктор Сергеевич, доктор технических наук, профессор, Волгоградский государственный технический университет, 400005, Российская Федерация, г. Волгоград, пр. Ленина, 28

В данной статье описывается управление стратегиями (эвристиками) распределения заданий в Грид-системе. Грид-система в рамках данной работы принимается состоящей из кластеров. Каждая кластерная система представлена в виде системы массового обслуживания (СМО), которая выполняет поступающие задания. Очередь полагается неограниченной из-за требования выполнить все задания. Стратегии позволяют балансировать нагрузку между кластерами Грид-системы. Рассматриваются детерминированная и стохастическая имитационные модели кластера и Грид-системы. Обе модели основаны на реальных записях параллельных нагрузок (представленных в виде SWF-логов), которые свободно доступны в Parallel Workloads Archive. В рамках данной статьи каждая нагрузка считается состоящей из трех значений. Каждая из них содержит время прихода задания, ширину и длину задания. Ширина – это количество вычислительных машин, которые необходимы для выполнения задания. Длина – это время выполнения задания на некоторой платформе. Площадь (сложность) является произведением длины и ширины. Предложены модели стохастической аппроксимации нагрузок с использованием распределений случайной величины. Дается список из десяти эвристик. Эти эвристики учитывают длину очереди, ее ширину, загрузку системы, использование ресурсов и т.п. Пользовательские оценки времени выполнения заданий крайне не точны по отношению к реальным временам выполнения. Поэтому мы не применяем пользовательские оценки. Детерминированная и стохастическая модели как кластерной, так и Грид-системы показывают аналогичный результат, поэтому мы можем предположить, что обе модели являются правильными. Если SWF-лог содержит некоторые аномалии, то результаты могут быть различными. Это означает, что информация из лога не в полной мере описывает поведение кластера. Разработанные модели нацелены на выбор наиболее эффективной стратегии распределения заданий.

¹ Работа выполнена при финансовой поддержке со стороны Минобрнауки России в рамках проектной части (проект № 2.1917.2014K_2014).