УДК 519.8 doi: 10.15421/321710

Е.М. Киселева, О.М. Притоманова, В.В. Шаравара, С.В. Журавель
Днипровский национальный университет им. Олеся Гончара

ОБЪЕКТНО-ОРИЕНТИРОВАННЫЙ ПОДХОД К ПРОГРАММНОЙ РЕАЛИЗАЦИИ АЛГОРИТМА РЕШЕНИЯ НЕЛИНЕЙНЫХ ЗАДАЧ ОПТИМАЛЬНОГО РАЗБИЕНИЯ МНОЖЕСТВ

Рассматривается применение парадигмы объектно-ориентированного программирования при программной реализации алгоритма решения нелинейных многопродуктовых задач оптимального разбиения множеств в случае выпуклой либо вогнутой нелинейной части целевого функционала с отысканием оптимальных координат центров подмножеств и ограничениями в форме равенств и неравенств, а также их частных случаев.

Розглядається застосування парадигми об'єктно-орієнтованого програмування при програмній реалізації алгоритму розв'язання нелінійних багатопродуктових задач оптимального розбиття множин у випадку опуклої або увігнутої нелінійної частини цільового функціоналу з відшуканням оптимальних координат центрів підмножин та обмеженнями у формі рівностей і нерівностей, а також їх частинних випадків.

Use of object-oriented paradigm for implementing the algorithm of solving nonlinear multigrocery problems of optimal partitioning of sets in case of convex or concave nonlinear component of target functional with optimal placing of the centers at restrictions in the form of equalities and inequalities and their particular cases is considered.

Ключевые слова: нелинейная задача оптимального разбиения множеств, недифференцируемая оптимизация, r-алгоритм, объектно-ориентированное программирование.

Введение. При программной реализации алгоритмов решения задач оптимального разбиения множеств (как и многих других математических задач) традиционно используется процедурный подход. С процедурным подходом связан ряд недостатков, таких как недостаточная переиспользуемость модулей, проблемы с инкапсуляцией данных и др., которые привели к переходу от процедурного программирования к объектно-ориентированному. Объектно-ориентированный подход значительно облегчает разработку и сопровождение сложных систем за счет того, что [3]:

- данные и связанные с ними операции объединены в одной сущности и локализованы в одном месте;
- инкапсуляция скрывает ненужные подробности реализации и защищает данные от несанкционированного доступа;
- возможно с легкостью получать сложные пользовательские типы данных, составляя их из более простых;

• повышается наглядность кода за счет использования соответствующих синтаксических возможностей языка.

Перечисленные плюсы объектно-ориентированного подхода привели к его повсеместному использованию.

В данной статье рассматривается применение указанного подхода при программной реализации алгоритма решения нелинейных многопродуктовых задач оптимального разбиения множеств в случае выпуклой либо вогнутой нелинейной части целевого функционала с отысканием оптимальных координат центров подмножеств и ограничениями в форме равенств и неравенств.

Постановка задачи. Пусть Ω – ограниченное, измеримое по Лебегу множество в n-мерном евклидовом пространстве E_n.

Найти

$$\min_{\left(\{\Omega_1^1,\ldots,\Omega_N^M\},\{\tau_1,\ldots,\tau_N\}\right)} \sum_{j=1}^{M}\sum_{i=1}^{N}[\varphi_i^j(\int_{\Omega_i^j}\rho^j(x)dx)+\int_{\Omega_i^j}c^j(x,\tau_i)\rho^j(x)dx]$$

при условиях:

$$\sum_{j=1}^{M}\int_{\Omega_i^j}\rho^j(x)dx=b_i, i=1,\ldots,p,\ \sum_{j=1}^{M}\int_{\Omega_i^j}\rho^j(x)dx\le b_i, i=p+1,\ldots,N,$$

$$\{\Omega_1^1,\ldots,\Omega_i^j,\ldots,\Omega_N^M\}\in\Sigma_\Omega^{NM},\ \tau=(\tau_1,\ldots,\tau_N)\in\Omega^N,$$

$$\sum\nolimits_{\Omega}^{NM} = (\{\Omega_1^1,\ldots,\ \Omega_i^j,\ldots,\ \Omega_N^M\}:\bigcup_{i=1}^{N}\Omega_i^j = \Omega,\ \mathrm{mes}(\Omega_i^j\cap\Omega_k^j)=0,$$

$$i\ne k,\ i,\ k=1,\ldots,N,\ j=1,\ldots,M).$$

Здесь функции $c^j(x,\tau_i)$ – действительные, ограниченные, определенные на $\Omega\times\Omega$, измеримые по аргументу x при любом фиксированном $\tau_i=(\tau_i^{(1)},\ldots,\tau_i^{(n)})$ из Ω, $i=1,\ldots,N$; функции $\rho^j(x)$ – действительные, ограниченные, измеримые и неотрицательные на Ω для всех $j=1,\ \ldots,\ M$; точка $\tau_i=(\tau_i^{(1)},\ldots,\tau_i^{(n)})\in\Omega$ – центр, общий для подмножеств $\Omega_i^1,\ldots,\Omega_i^M$, $i=1,\ldots,N$, один и тот же для всех $j=1,\ \ldots,\ M$; $\varphi_i^j(Y_i^j)$, $i=1,\ldots,N, j=1,\ldots,M$ – действительные, ограниченные, как выпуклые, так и вогнутые, дважды непрерывно дифференцируемые функции своего аргумента, где $Y_i^j=\int_{\Omega_i^j}\rho^j(x)dx$; $b_1,\ldots,b_N$ – заданные неотрицательные числа, причем

$$S=\int_{\Omega}\sum_{j=1}^{M}\rho^j(x)dx\le\sum_{i=1}^{N}b_i,\ 0\le b_i\le S,\ \ i=1,\ldots,N.$$

Метод решения. Для решения рассматриваемой нелинейной задачи оптимального разбиения множеств был использован алгоритм, описанный в работе [1]. Идея алгоритма заключается в следующем.

Исходная бесконечномерная задача оптимизации сводится через функционал Лагранжа к вспомогательной двойственной конечномерной негладкой задаче максимина, для численного решения которой применяется модификация r-алгоритма Шора (метод обобщенных псевдоградиентов с растяжением пространства в направлении разности двух последовательных псевдоградиентов) [2].

Особенностью алгоритма решения вышеназванной задачи является получение оптимального решения аналитически в виде некоторого операторного уравнения с параметрами, отыскиваемыми как оптимальное решение конечномерной негладкой задачи оптимизации.

Особенности программной реализации. Для программной реализации использовался объектно-ориентированный язык программирования C# и среда разработки Microsoft Visual Studio. На рис. 1 и 2 представлена схема основных классов, использующихся в вычислительной составляющей программы.

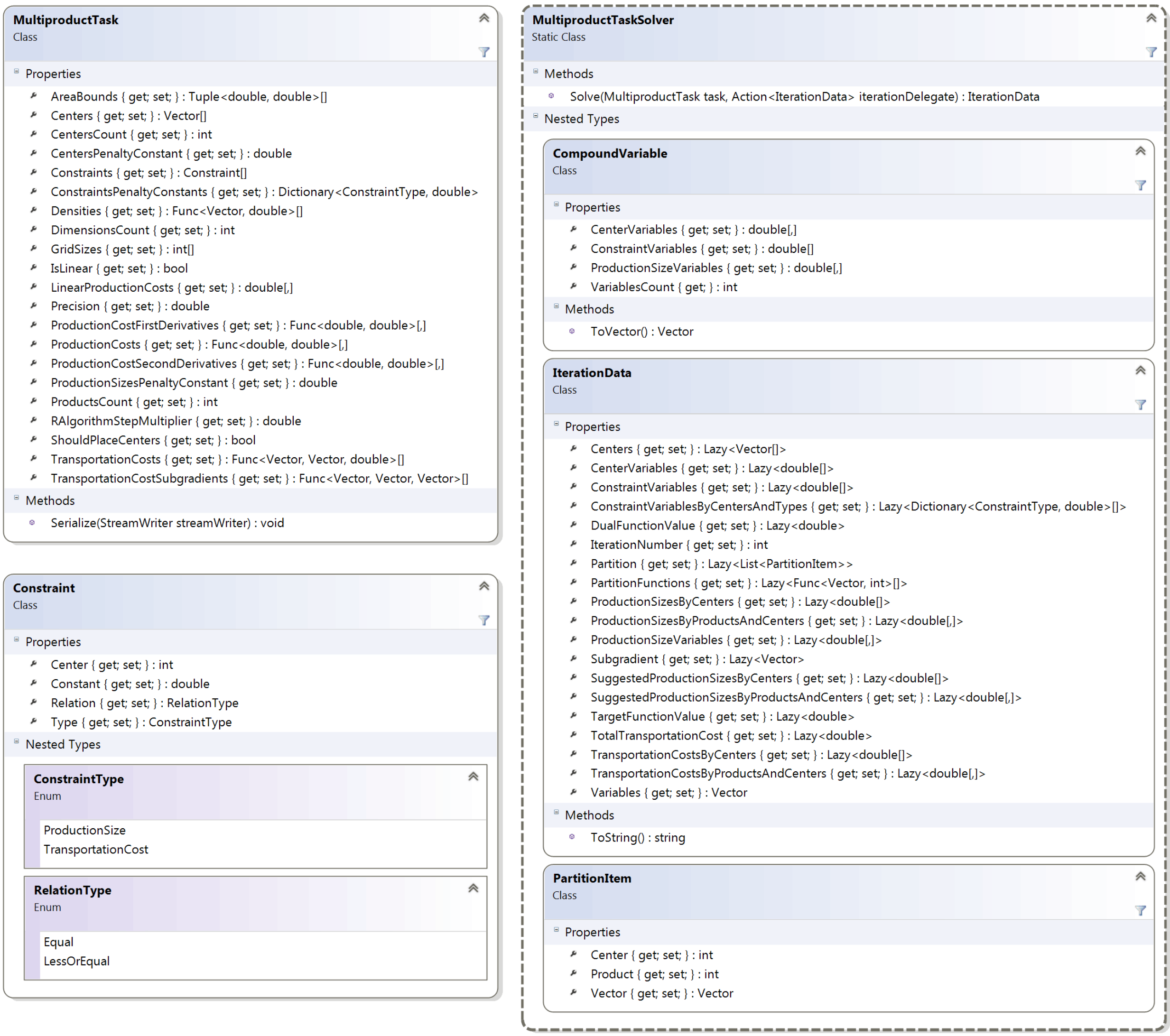

Рис. 1. Диаграмма классов (часть 1)

Вычислительная схема состоит из следующих этапов:

Этап 1. Формирование входных данных.

На этом этапе формируется объект типа MultiproductTask, который содержит все исходные параметры решаемой задачи:

- int DimensionsCount { get; set; } – размерность n пространства E_n;

• Tuple<double, double>[] AreaBounds { get; set; } – границы области Ω по каждому из измерений;

• int[] GridSizes { get; set; } – количество узлов сетки, которой покрывается область Ω, по каждому из измерений;

• int ProductsCount { get; set; } – количество продуктов *M*;

• Func<Vector, Vector, double>[] TransportationCosts { get; set; } – массив функций $c^j(x,\tau),\, j=1,\ldots,M$;

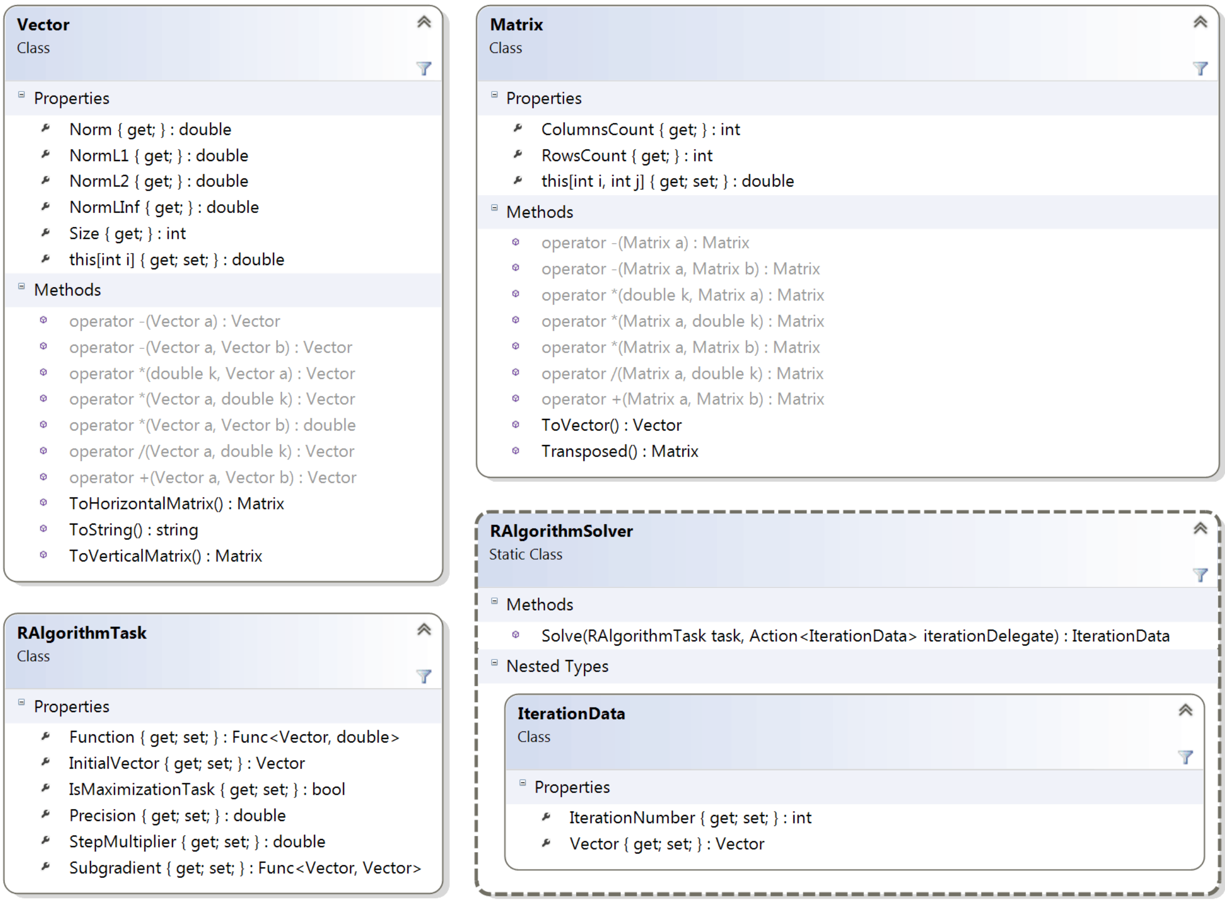

Рис. 2. Диаграмма классов (часть 2)

• Func<Vector, double>[] Densities { get; set; } – массив функций $\rho^j(x),\, j=1,\ldots,M$;

• bool ShouldPlaceCenters { get; set; } – указывает, нужно ли размещать центры подмножеств;

• int CentersCount { get; set; } – количество подмножеств *N*;

• Vector[] Centers { get; set; } – координаты центров подмножеств $\tau_i=(\tau_i^{(1)},\ldots,\tau_i^{(n)})\in\Omega, i=1,\ldots,n$;

• bool IsLinear { get; set; } – указывает, является ли задача линейной;

• double[,] LinearProductionCosts { get; set; } – матрица $a_i^j, i=1,\ldots,N,\, j=1,\ldots,M$ (в случае линейной задачи);

• Func<Double, Double>[,] ProductionCosts { get; set; } – массив функций $\varphi_i^j(Y_i^j), i=1,\ldots,N,\, j=1,\ldots,M$;

- Func<Double, Double>[,] ProductionCostFirstDerivatives { get; set; } – массив производных функций $\varphi_i^j(Y_i^j), i=1,...,N, j=1,...,M$;
- Func<Double, Double>[,] ProductionCostSecondDerivatives { get; set; } – массив вторых производных функций $\varphi_i^j(Y_i^j), i=1,...,N, j=1,...,M$;
- Constraint[] Constraints { get; set; } – массив ограничений.

Этап 2. Передача входных данных.

Сформированный объект типа MultiproductTask передается на вход методу Solve класса MultiproductTaskSolver. В ходе выполнения этого метода формируется объект типа RAlgorithmTask, который содержит входные данные для r-алгоритма:

- Vector InitialVector { get; set; } – начальное приближение для переменных τ_i, двойственных переменных ψ_i и переменных Y_i^j;
- Func<Vector, double> Function { get; set; } – целевая функция;
- Func<Vector, Vector> Subgradient { get; set; } – субградиент целевой функции;
- double StepMultiplier { get; set; } – начальное значение шагового множителя;
- double Precision { get; set; } – точность, которую необходимо достигнуть;
- bool IsMaximizationTask { get; set; } – указывает максимизировать или минимизировать значение целевой функции.

Сформированный объект типа RAlgorithmTask передается на вход методу Solve класса RAlgorithmSolver.

Этап 3. Выполнение *r*-алгоритма.

На данном этапе, согласно итерационным формулам *r*-алгоритма [2], производится поиск оптимума переданной функции. После каждой итерации формируется объект типа RAlgorithmSolver.IterationData, который содержит результаты итерации *r*-алгоритма:

- int IterationNumber { get; set; } – номер итерации;
- Vector Vector { get; set; } – текущее приближение.

Сформированный объект передается в callback-функцию, переданную в метод Solve класса RAlgorithmSolver, где на его основе формируется объект типа MultiproductTaskSolverIterationData, который содержит информацию о текущей итерации:

- int IterationNumber { get; set; } – номер итерации;
- Vector Variables { get; set; } – вектор всех переменных;
- Lazy<double[]> ConstraintVariables { get; set; } – значения двойственных переменных ψ_i;
- Lazy<Vector[]> Centers { get; set; } – центры подмножеств;
- Lazy<double[,]> ProductionSizeVariables { get; set; } – значения переменных Y_i^j;
- Lazy<Vector> Subgradient { get; set; } – вектор субградиента;

- Lazy<double> TargetFunctionValue { get; set; } – значение функционала исходной задачи;
- Lazy<double> DualFunctionValue { get; set; } – значение функционала двойственной задачи;
- Lazy<double[,]> ProductionSizesByProductsAndCenters { get; set; } – значения величин $\int\limits_{\Omega}\sum\limits_{j=1}^{M}\rho^j(x)\lambda_i^j(x)dx, i=1,...,N, j=1,...,M$;
- Lazy<double[,]> TransportationCostsByProductsAndCenters { get; set; } – значения величин $\int\limits_{\Omega}\sum\limits_{j=1}^{M}\rho^j(x)c^j(x,\tau_i)\lambda_i^j(x)dx, i=1,...,N, j=1,...,M$;
- Lazy<List<PartitionItem>> Partition { get; set; } – разбиение.

Сформированный объект передается в callback-функцию, переданную в метод Solve класса MultiproductTaskSolver, где эти объекты сохраняются для предоставления возможности просмотра результатов каждой итерации по окончании процесса.

После выполнения критерия остановки процесс вычисления останавливается и происходит отображение результатов последней итерации.

Визуальная составляющая программы использует следующие классы:

- FormInput – форма, которая используется для ввода параметров задачи, а также их сохранения и загрузки из файла;
- FormOutput – форма, которая используется для отображения результатов;
- PartitionControl – контрол, который отвечает за визуальное отображение разбиения по фиксированному продукту.

Анализ результатов. Работа разработанной программы была проверена на модельных задачах.

Модельная задача 1.

$$\Omega=\{(x,y):0\le x\le 10, 0\le y\le 10\},\ N=9,\ M=3,$$

$$c^j(x,y,\tau_i)=\begin{cases}\sqrt{(x-\tau_i^{(1)})^2+(y-\tau_i^{(2)})^2}, & \text{если } j=1,\\ \max\left(\left|x-\tau_i^{(1)}\right|,\left|y-\tau_i^{(2)}\right|\right), & \text{если } j=2,\\ \left|x-\tau_i^{(1)}\right|+\left|y-\tau_i^{(2)}\right|, & \text{если } j=3,\end{cases}$$

$$\rho^j(x,y)=\frac{1}{\ln\left|(x-y)^j-110.003\right|},\ j=\overline{1,\ 3},$$

$$0\le\sum_{j=1}^{3}\iint\limits_{\Omega_i^j}\rho^j(x,y)dxdy\le b_i,\ \ i=1,\ 2,\ 4,\ 5,\ 7,\ 9,$$

$$b_{1,7}=100,\ b_2=86,\ b_4=80,\ b_5=17,\ b_9=25,$$

$$\sum_{j=1}^{3}\iint\limits_{\Omega_i^j}\rho^j(x,y)dxdy=b_i,\ \ i=3,\ 6,\ 8,$$

$$b_3 = 36\,,\; b_6 = 5\,,\; b_8 = 15\,.$$

$$\varphi_i^j(Y_i^j) = (Y_i^j)^2\,,\; i = \overline{1,\ 9}\,.$$

Полученные результаты:

- значение прямого функционала: 669,7,
- значение двойственного функционала: 671,4.

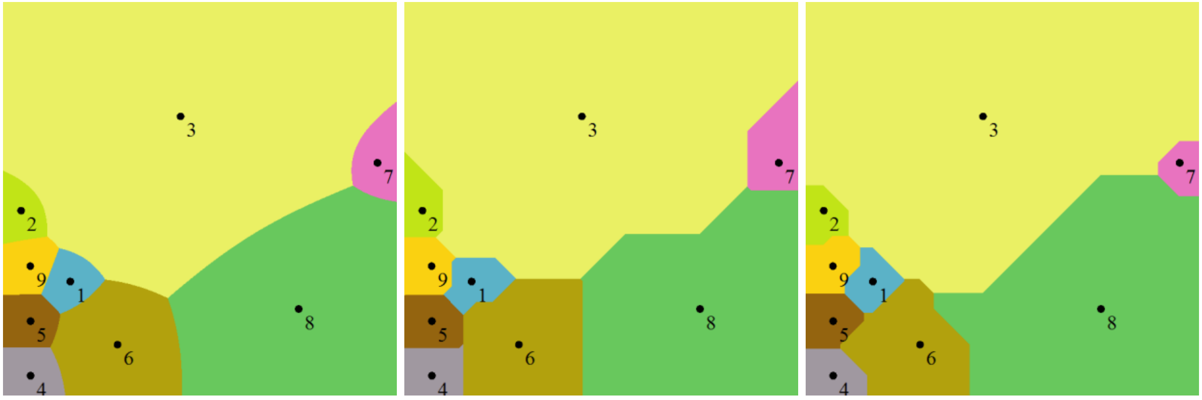

Рис. 3. Оптимальные разбиения по 1-му (слева), 2-му (посередине) и 3-му (справа) продукту, полученные для модельной задачи 1

Результаты, полученные в [1], для модельной задачи 1 с помощью программы, выполненной на Visual Fortran:

- значение прямого функционала: 676,4,
- значение двойственного функционала: 673,4.

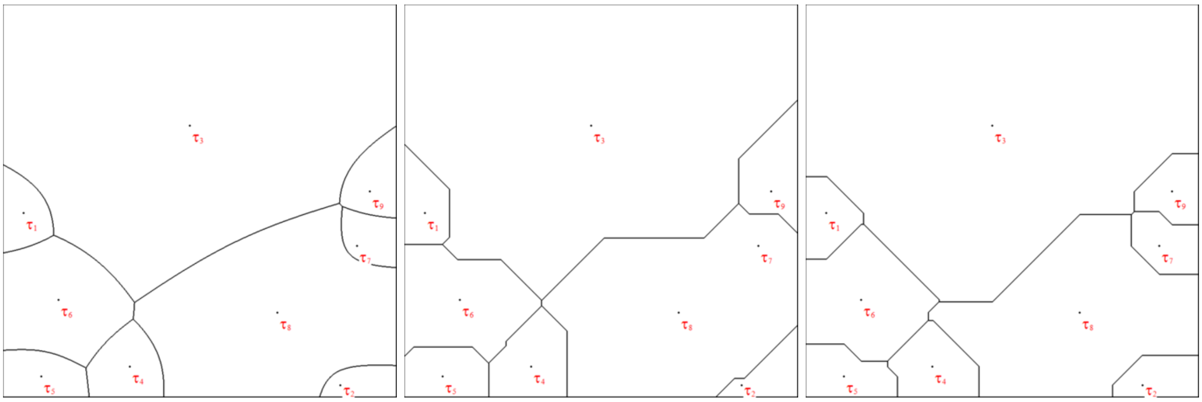

Рис. 4. Оптимальные разбиения по 1-му (слева), 2-му (посередине) и 3-му (справа) продукту, полученные в [1] для модельной задачи 1

Модельная задача 2. Все параметры такие же, как в модели 1, за исключением $\rho^j(x,y) = 1,\ j = \overline{1,\ 3}$.

Полученные результаты:

- значение прямого функционала: 4682,9,
- значение двойственного функционала: 4680,5.

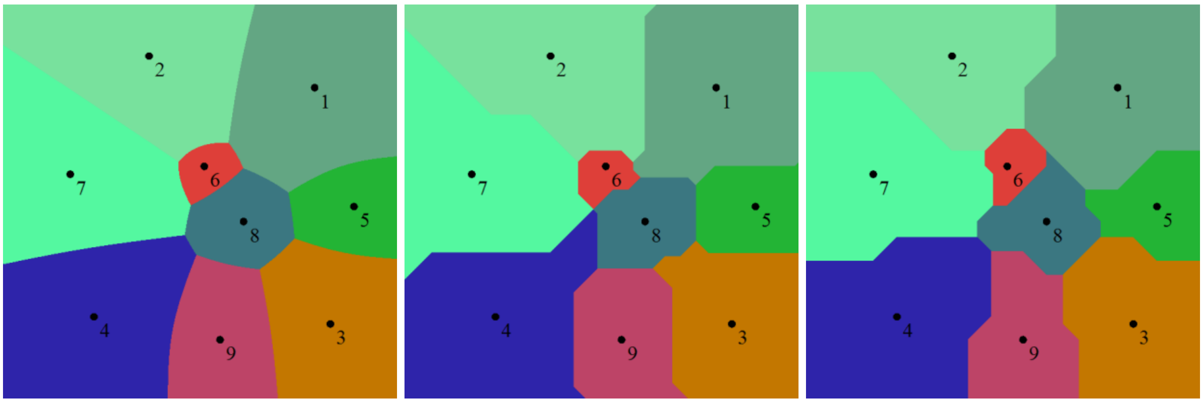

Рис. 5. Оптимальные разбиения по 1-му (слева), 2-му (посередине) и 3-му (справа) продукту, полученные для модельной задачи 2

Результаты, полученные в [1], для модельной задачи 2 с помощью программы, выполненной на Visual Fortran:

- значение прямого функционала: 4637,8,
- значение двойственного функционала: 4676,7.

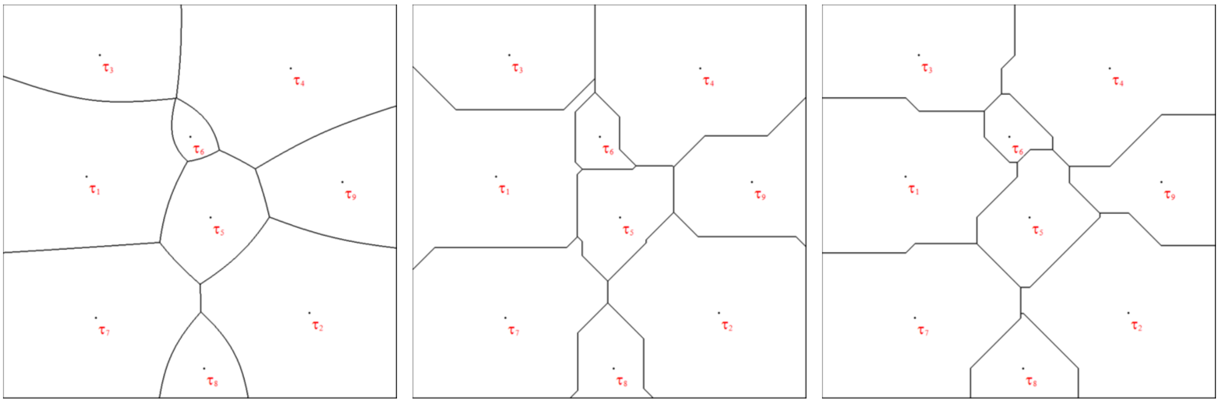

Рис. 6. Оптимальные разбиения по 1-му (слева), 2-му (посередине) и 3-му (справа) продукту, полученные в [1] для модельной задачи 2

Полученные результаты совпадают с результатами, представленными в [1], что подтверждает корректность работы разработанного программного обеспечения.

Выводы. Разработано программное обеспечение, позволяющее отыскивать решения нелинейных многопродуктовых задач оптимального разбиения множеств в случае выпуклой либо вогнутой нелинейной части целевого функционала с отысканием оптимальных координат центров подмножеств и ограничениями в форме равенств и неравенств. Ключевой особенностью программного обеспечения является объектно-ориентированный подход к программированию. Разработанное программное обеспечение при задании соответствующих входных данных позволяет также решать задачи, являющиеся частными случаями рассматриваемой, а именно: линейные, однопродуктовые, задачи с фиксированными координатами центров подмножеств, задачи без ограничений.

Библиографические ссылки

1. **Киселева, Е.М.** Модели и методы решения непрерывных задач оптимального разбиения множеств: линейные, нелинейные, динамические задачи: монография [Текст] / Е.М. Киселева, Л.С. Коряшкина – К.: Наукова думка, 2013. – 606 с.
2. **Киселева, Е.М.** Непрерывные задачи оптимального разбиения множеств: монография [Текст] / Е.М. Киселева, Н.З. Шор. – К.: Наукова думка, 2005. – 564 с.
3. **Troelsen, A.** Pro C# and the .NET 4.5 Framework, Sixth Edition [Text] / A. Troelsen. – NY: Springer Science+Business Media, 2012. – 1487 c.

Надійшла до редколегії: 30.03.2017