СЕКЦИЯ 2. СИСТЕМЫ АВТОМАТИЗАЦИИ И ИНФОРМАТИЗАЦИИ В ОБРАЗОВАНИИ, НАУКЕ И ПРОИЗВОДСТВЕ

УДК 338.984

Г. С. Алексеев, В. Ю. Носков
ФГАОУ ВО «Уральский федеральный университет
имени первого Президента России Б.Н. Ельцина», г. Екатеринбург, Россия

РЕШЕНИЕ ЗАДАЧ ЛИНЕЙНОГО ПРОГРАММИРОВАНИЯ С ИСПОЛЬЗОВАНИЕМ MICROSOFT SOLVER FOUNDATION НА ПРИМЕРЕ ОПТИМАЛЬНОГО РАСПРЕДЕЛЕНИЯ ТОПЛИВНЫХ РЕСУРСОВ В ГРУППЕ ДОМЕННЫХ ПЕЧЕЙ

Аннотация

Статья посвящена вопросам программной реализации решения задач линейного математического программирования с использованием нового инструмента от корпорации Microsoft – Solver Foundation. Освещены основные положения при использовании данного инструмента, разобраны соответствующие примеры.

Ключевые слова: *линейное программирование, microsoft solver foundation, математические модели, линейная оптимизация.*

Abstract

Article is devoted to the software implementation of solving linear mathematical programming problems using a new tool from Microsoft Corporation – Solver Foundation. Highlights the main provisions when using this tool, analyzed relevant examples.

Key words: *linear programming, microsoft solver foundation, mathematical models, optimization.*

Практически все технические и экономические задачи так или иначе сводятся к линейным математическим моделям, встает вопрос, как быстро и с высокой точностью их рассчитывать. В век компьютерных технологий человеку с решением этой непростой задачи способны помочь машины.

Сформулируем техническую задачу о распределении топливных ресурсов в группе доменных печей.

В состав доменного цеха входит N печей, на каждой из которых осуществляется инжектирование природного газа (ПГ) с расходами $V_i^{\text{пг}}$ ($i = 1, \ldots, N$), м^3/ч. Эффективность инжектируемого топлива оценивается эквивалентом замены кокса e_i (кг/м^3), физический смысл которого – это количество кокса, сэкономленного на i-й печи при вдувании 1 м^3 природного газа. Задан лимит на подачу природного газа в цех $V_\Sigma^{\text{пг}}$ (м^3/час), а также прямые ограничения по расходу газа на каждую доменную печь, в частности, $V_{i,min}^{ПГ} \le V_i^{ПГ} \le V_{i,max}^{ПГ}$. Установлены также цеховой ресурс по расходу кокса и требуемая производительность доменного цеха по чугуну, которые равны соответственно $К_\Sigma$ (т кокса/ч) и $П_\Sigma$ (т чугуна/ч). В качестве технологического ограничения на

каждую из доменных печей приняты температурные интервалы $T_{i,max} - T_{i,min}$, в пределах которых должны находиться температуры горения на фурмах T_i, характеризующие тепловое состояние доменных печей. Известны стоимости кокса и природного газа, которые соответственно равны $C_к$ (руб/кг) и $C_{пг}$ (руб/м^3).

Требуется распределить подачу природного газа между печами доменного цеха таким образом, чтобы при существующих ограничениях была достигнута максимальная экономия кокса в целом по цеху.

Математическая модель задачи. Экономия кокса в целом по цеху при перераспределении природного газа в группе доменных печей (целевая функция) имеет вид

$$Z = \sum_{i=1}^{N} (e_i C_к - C_{ПГ}) \cdot V_i^{ПГ} \to \max_{V_i^{ПГ}}, \quad (1)$$

Здесь N – число печей в рассматриваемой группе (цехе); e_i – эквивалент замены кокса, кг кокса/(м^3 ПГ); $C_к$ – стоимость кокса, руб/кг; $C_{ПГ}$ – стоимость природного газа, руб/м^3; $V_i^{ПГ}$ – расход природного газа на i-й доменной печи, м^3/ч.

Ограничения:

- по расходу природного газа в целом по цеху

$$\sum_{i=1}^{N} V_i^{ПГ} \le V_\Sigma^{ПГ}, \quad (2)$$

где $V_\Sigma^{ПГ}$ – резерв по расходу природного газа в целом по цеху, м^3/ч.

- по расходу кокса в целом по цеху

$$\sum_{i=1}^{N} \left\{ K_{0i} + 0{,}001 \left(V_{i0}^{ПГ} - V_i^{ПГ} \right) e_i \right\} \le K_\Sigma, \quad (3)$$

где K_{0i} – расход кокса на i-ю печь в базовом периоде, т/ч; K_Σ – запасы кокса по цеху, т/ч; $V_{i0}^{ПГ}$ – расход природного газа на i-й печи в базовом периоде, м^3/ч.

- по объему производства чугуна в целом по цеху

$$\sum_{i=1}^{N} \left\{ \left(V_i^{ПГ} - V_{i0}^{ПГ} \right) \Delta П_i^{ПГ} - e_i \left(V_i^{ПГ} - V_{i0}^{ПГ} \right) \Delta П_i^{К} + П_{i0} \right\} \ge П_\Sigma, \quad (4)$$

где $П_\Sigma$ – требуемое производство чугуна в цехе, т/ч; $П_{i0}$ – производительность i-й печи по чугуну, т/ч; $\Delta П_i^{ПГ}$ – прирост производства чугуна при изменении расхода природного газа на i-й печи, т чуг/(м^3 ПГ); $\Delta П_i^{К}$ – прирост производства чугуна при изменении расхода кокса на i-й печи, т чуг/(кг кокса).

- по расходу природного газа на каждую из печей

$$V_i^{ПГ} \ge V_{i\min}^{ПГ}; \quad V_i^{ПГ} \le V_{i\max}^{ПГ}, \quad (5)$$

где $V_{i\min}^{ПГ}, V_{i\max}^{ПГ}$ – соответственно минимально и максимально допустимые расходы природного газа на i-ю доменную печь, м^3/ч.

- по температуре горения на фурмах на каждую из печей

$$\begin{cases} \left(V_i^{ПГ} - V_{i0}^{ПГ}\right)\Delta T_i^{ПГ} + \mathrm{T_{i0}} \le T_{i\,max}; \\ \left(V_i^{ПГ} - V_{i0}^{ПГ}\right)\Delta T_i^{ПГ} + \mathrm{T_{i0}} \ge T_{i\,min} \end{cases}, \quad (6)$$

где $\Delta T_i^{ПГ}$ – изменение теоретической температуры горения при увеличении расхода природного газа на *i*-й печи, °С/(м^3 ПГ); T_{i0} – теоретическая температура горения на фурмах *i*-й доменной печи, °С; $T_{i\min}, T_{i\max}$ – соответственно минимально и максимально допустимые температуры горения на фурмах *i*-й доменной печи, °С.

Данную задачу с применением компьютерной техники можно решить несколькими путями: используя специализированные математические пакеты (MathCad, Matlab, Maple), с использованием табличного редактора Excel или с помощью вновь разработанного приложения под конкретную задачу в конкретной предметной области.

Математические пакеты и табличный редактор удобно использовать «здесь и сейчас», то есть рассчитать какую-нибудь математическую модель, записать результаты и закрыть программу. Однако такой подход неприменим, когда речь идет о прикладном программном обеспечении, задачей которого является установление оптимального соотношения расхода природного газа в доменном цехе. Обычным сотрудникам, использующим прикладное ПО, гораздо удобнее, чтобы расчеты производились не в математическом пакете, для работы с которым требуются специальные знания, а в удобной настольной программе или веб-приложении, с удобным пользовательским интерфейсом.

Ранее при необходимости создать ПО такого типа прикладным программистам зачастую приходилось реализовывать решения математических задач самостоятельно (заново программируя реализации алгоритмов) или создавать связи между создаваемым приложением и программой, производящей расчет.

По мере того, как усложнялось прикладное программное обеспечение, написанное таким образом, становилось понятно, что лишние зависимости в виде обращений к сторонним программам не несут пользы и необходимы специализированные библиотеки для решения подобного класса задач с использованием языка программирования, нативно, без обращений к API другого ПО.

В 2011 году была выпущена первая версия библиотека Microsoft Solver Foundation для .NET разработчиков. Solver Foundation – это библиотека для математического программирования, моделирования и оптимизации. С помощью математического моделирования решаются задачи принятия решений. Solver Foundation предлагает .NET-разработчикам высококачественные инструменты для оптимизации их решений, позволяя решать модели в приложениях даже разработчикам, которые не являются экспертами в математическом моделировании.

На сегодняшний момент Solver Foundation обладает следующими ключевыми возможностями:

– моделирование и решение сценариев с помощью ограничений, целей и данных;

– разработка на языке Optimization Modeling Language (OML), императивно в C#, функционально в F# или на любом другом языке .NET;

– встроенные решатели задач для наиболее распространенных типов моделей;

– интеграция с популярными решателями: Gurobi, Ziena Knitro, Frontline Solver Platform SDK, Mosek, FICO Xpress, LINDO и lp_solve;

– интеграция с популярными инструментами Microsoft Office Excel и SharePoint для создания и решения моделей.

Основные операции при работе с MSF:

1. Создание контекста MSF и модели решения задачи.

```
var solver = SolverContext.GetContext();
var model = solver.CreateModel();
```

2. Добавление расчетного параметра (изменяемого значения).

```
var decisionX = new Decision(Domain.IntegerNonnegative, "X");
model.AddDecision(decisionX);
```

3. Добавление ограничений.

```
model.AddConstraint("Constraint0", 100 <= decisionX);
```

4. Определение целевой функции, ее типа.

```
model.AddGoal("Goal", GoalKind.Maximize, (-2 * decisionX) + (5 * decisionY));
```

5. Получение решения.

```
var solution = solver.Solve();
double x = decisionX.GetDouble();
```

Приведем упрощенный листинг кода, реализующий данный алгоритм действий применительно к задаче о распределении топливных ресурсов.

```
var furnances = optimizationModel.Furnaces;
SolverContext context = SolverContext.GetContext();
Model model = context.CreateModel();

var furnancesSet = new Set(Domain.Any, "Furnances");
var parametersList = new List<Parameter>();
var gasExpenseBasePeriod = new Parameter(Domain.Real, "GasExpenseBasePeriod", furnancesSet);
gasExpenseBasePeriod.SetBinding(furnances, "GasExpenseBasePeriod", "FurnanceId");

//...
parametersList.Add(gasExpenseBasePeriod);
//...
model.AddParameters(parametersList.ToArray());

Decision decision = new Decision(Domain.RealNonnegative, "decision", furnancesSet);
model.AddDecision(decision);

/*(koksekv*стоимость кокса - стоимость природного газа)*расход ту readjust*/
model.AddGoal("CokeSaving", GoalKind.Maximize,
    Model.Sum(Model.ForEach(furnancesSet,  FurnanceId =>
    (coxReplacementEquivalent[FurnanceId] * optimizationModel.CoxCost - optimizationModel.GasCost)
                                                    * decision[FurnanceId])));
model.AddConstraint("RequiredIronPerformance",
    Model.Sum(Model.ForEach(furnancesSet, CalculatePerformanceConstraint)) >= optimizationModel.RequiredIronPerformance);

// (расходПГ - расходБазовомПериоде)*ИзмПрЧугИзмПГ - ЭквЗамКокс*
//(расходПГ - расходБазовомПериоде)*ИзмПрЧугИзимКокса + ПрПоЧугБазовомПер
Term CalculatePerformanceConstraint(Term FurnanceId)
{
    var val = (decision[FurnanceId] - gasExpenseBasePeriod[FurnanceId]) * deltaIronPerformanceGasChanged[FurnanceId] -
              coxReplacementEquivalent[FurnanceId] * (decision[FurnanceId] - gasExpenseBasePeriod[FurnanceId]) *
              deltaIronPerformanceCoxChanged[FurnanceId] + ironPerformance[FurnanceId];
    return val;
}
//..
SimplexDirective simplex = new SimplexDirective();
Solution solved = context.Solve(simplex);
var report = solved.GetReport();
```

Рис. 1. Фрагмент программной реализации задачи оптимизации

Можно подытожить, что Microsoft Solver Foundation – это мощный набор математических инструментов, позволяющий решать математические задачи при написании приложений. Фремворк имеет достаточно простой API, из этого следует, что для начала работы с ним не потребуется каких-либо дополнительных знаний, кроме как программирование на C#.

Список использованных источников

1. Лунгу К.Н. Линейное программирование. Руководство к решению задач. – М.: ФИЗМАТЛИТ, 2005. – 128 с.

2. Агальцов В.П. Математические методы в программировании / В.П. Агальцов. – М.: Форум, 2010. – 240 с.

3. Иен Г. Программирование на C# 5.0 / Г. Иен. – М.: Эксмо, 2014. – 1136 с.

УДК 004.41

А. Е. Аптышев, В. А. Гольцев
ФГАОУ ВО «Уральский федеральный университет
имени первого Президента России Б.Н. Ельцина», г. Екатеринбург, Россия

РАЗРАБОТКА ПРОГРАММНОГО ПРОДУКТА ДЛЯ РАСЧЕТА РАССЕИВАНИЯ ВРЕДНЫХ ВЕЩЕСТВ В АТМОСВЕРЕ ОТ МЕДЕПЛАВИЛЬНОГО ЦЕХА АО «УРАЛЭЛЕКТРОМЕДЬ»

Аннотация

Данная работа посвящена созданию программного продукта для расчета рассеивания вредных веществ в атмосфере от медеплавильного цеха АО «Уралэлектромедь» и несет в себе практическую значимость в задачах расчета. Во введении обосновывается актуальность выбранной темы, формируются цель и задачи исследования, указывается объект и предмет исследования. Рассматривается такая задача, как сбор данных, их обработка и контрольный расчет. Сбор данных является первоочередной задачей, так как они необходимы для выполнения расчета и получения конечного результата, который являются основными показателями работы. Автор предоставляет программный продукт для расчета рассеивания вредных веществ в атмосфере. Приложение поможет рассчитать концентрацию вредных веществ в атмосфере, их разброс, а также рассчитать опасную скорость ветра.

Ключевые слова: *сбор данных, windows – приложение, расчет рассеивания вредных веществ, АО «Уралэлектромедь», c#.*

Abstract

This work is devoted to the creation of a software product for calculating the dispersion of harmful substances in the atmosphere from the copper smelter of JSC "Uralelektromed" and carries practical significance in the problems of calculation. In the introduction, the relevance of the chosen topic is justified, the purpose and objectives of the study are formed, the object and subject of the study are indicated. The problem of data collection, processing and control calculation is considered. Data collection is a priority, as it is necessary for the final calculation, the results of which are the