The Problem of Mixed Integer Programming for Optimal Schedule of Petroleum Products Manufacturing

Olga N. Kaneva
Omsk State Technical University
Omsk, Russia,
Mira Ave. 11, 644050
onkaneva@omgtu.ru

Anna V. Zykina
Omsk State Technical University
Omsk, Russia,
Mira Ave. 11, 644050
avzykina@mail.ru

Maria M. Volodchenko
Omsk State Technical University
Omsk, Russia,
Mira Ave. 11, 644050
marymind99@gmail.com

Abstract

A mathematical model of commercial production of petroleum products is considered, with an optimal control being the one minimizing deviations from the target formulation. The resulting optimization problem of mixed integer programming depends on two types of variables:

1) the flow rate of petroleum product;

2) a discrete value that takes the values 0 or 1, which characterizes the start of the certification in a given time interval in a given commercial tank before shipment to a given receiving object.

To solve the problem of mixed integer programming, the following approach is proposed: at each iteration, the value of the certification vector is found using a genetic algorithm and registered. Then, by solving the optimization problem, the optimal control is determined, i.e. the flow rate of the petroleum product. A computational experiment is carried out to calculate the optimal schedule for the proposed process flow chart.

1 Introduction

Commercial production of petroleum products (gasolines) is carried out by mixing various components in the proportions specified by the formulation. The operation schedule of the oil plant equipment should streamline the components mixing process and be made for a given calendar plan. At the same time, it should take into account the current state of the equipment and minimally deviate from the estimated production schedule for a given period. For the effective operation of oil refining enterprises, it is necessary to coordinate the dispatching schedule with the actual state of production [Hus2021, Zyk2018, Zyk018, Sav2018, Zyk2019].

Petroleum products manufacturing is complicated by the following factors:

– if more than one brand of gasoline is needed and all components are available, it is necessary to mix them so that there are no residues;

– it is necessary to take into account the change in the operating modes of some installations;

In: Sergei S. Goncharov, Yuri G. Evtushenko (eds.): Proceedings of the Workshop on Applied Mathematics and Fundamental Computer Science 2021 , Omsk, Russia, 24-04-2021, published at http://ceur-ws.org

– it is necessary to take into account the distribution of incoming and outgoing flows.

Besides:

– the composition of crude oil may change;

– needs and prices may change;

– depending on uncertain factors (the time passing after the reactor shutdown, the activity of the catalyst, the air temperature, the temperature of the cooling water, etc.), the yields of petroleum products may change.

The schedule should be determined dynamically following the updated calendar task and changes in the actual state of the equipment. Changes can occur at any time, and the dispatching planning system must either find an effective solution or inform about the impossibility of performing an updated calendar task.

The dispatching schedule should be calculated since the components have to be included in the production more fully according to the total amount of the components arriving for commercial production for the entire period of the calendar schedule, provided that all specified restrictions are met. Such an optimal dispatching schedule will ensure a more coordinated operation of the refinery both for the period of the dispatching schedule and for the period of the calendar plan.

At the moment, owing to technical development, it is possible to create and implement advanced control systems for petroleum products manufacturing. The proposed algorithm for calculating the optimal dispatching schedule can be the basis of such a system that combines the work of specialists at different levels of business.

2 Mathematical model of the problem

The first step in solving the problem of optimal scheduling is to formalize the blending processes in the form of a mathematical model. Consider the following description of the parameters of the mathematical model [Zyk2020, Sav2019, Zyk019].

$Types = \{pipe, pump, tank, pipe_c, tank_g, stock\}$ is a set of values that characterize the type of an object, the numbers of its elements corresponding to the numbers of the object:

$-pipe$ is the type of object that represents the process piping in the diagram;

$-pump$ is the type of object that represents a pumping station;

$-tank$ is the type of an object that represents a tank;

$-pipe_c$ is the type of object that represents a components' blending unit;

$-tank_g$ is the type of object that represents a commercial tank;

$-stock$ is the type of object that represents an outflow.

Constant parameters:

$-Range$ is the period for which the dispatching schedule is being made;

$-N_{products}$ is the amount of the end products;

$-N_{inputs}$ is the number of objects that are sources of components, $p = 1, ..., N_{inputs}$

$-N_{stock}$ the number of receiving objects for specific end products, $N_{stock} = N_{products}$, $s = 1, ..., N_{stock}$;

$-N_{tank_g}$ is the number of commercial tanks, $g = 1, ..., N_{tank_g}$;

$-N_{pipes}$ is the number of pipes, $b = 1, ..., N_{pipes}$;

$-N_{tanks}$ is the number of component tanks, $d = 1, ..., N_{tanks}$;

$-N_{times}$ is the number of time intervals, $t = 1, ..., N_{times}$;

$-N_{objects}$ is the number of objects involved in the model:

$$N_{objects} = N_{inputs} + N_{pipes} + N_{tank_g} + N_{stock}, i = 1, ..., N_{objects};$$

.

$-N_l$ is the number of connections between the model objects (equipment), $l = 1, ..., N_{objects}$.

The desired parameters:

– Matrix F is the matrix of flow rates through the connections in time intervals:

$$\begin{pmatrix} f_1^1 & f_1^2 & \dots & f_1^t & \dots & f_1^{N_{times}} \\ f_2^1 & f_2^2 & \dots & f_2^t & \dots & f_2^{N_{times}} \\ \dots & \dots & \dots & \dots & \dots & \dots \\ f_l^1 & f_l^2 & \dots & f_l^t & \dots & f_l^{N_{times}} \\ \dots & \dots & \dots & \dots & \dots & \dots \\ f_{N_l}^1 & f_{N_l}^2 & \dots & f_{N_l}^t & \dots & f_{N_l}^{N_{times}} \end{pmatrix}$$

f_l^t is the rate of the flow passing through the l-th connection for the t-th interval, $l = 1, ..., N_l$, $t = 1, ..., N_{times}$

– The matrix $Pass$:

$$\begin{pmatrix} Pass^1 \\ Pass^2 \\ \dots \\ Pass^s \\ \dots \\ Pass^{N_{stock}} \end{pmatrix}$$

$Pass^s$ is a matrix in which the columns are the time *intervalt*, and the rows are the connections linking the commercial tanks $g = 1, ..., N_{tank_g}$ with the outflow s:

$$\begin{pmatrix} {Pass^s}_1^1 & {Pass^s}_1^2 & \dots & {Pass^s}_1^t & \dots & {Pass^s}_1^{N_{times}} \\ {Pass^s}_2^1 & {Pass^s}_2^2 & \dots & {Pass^s}_2^t & \dots & {Pass^s}_2^{N_{times}} \\ \dots & \dots & \dots & \dots & \dots & \dots \\ {Pass^s}_l^1 & {Pass^s}_l^2 & \dots & {Pass^s}_l^t & \dots & {Pass^s}_l^{N_{times}} \\ \dots & \dots & \dots & \dots & \dots & \dots \\ {Pass^s}_{N_{tanks_g}}^1 & {Pass^s}_{N_{tanks_g}}^2 & \dots & {Pass^s}_{N_{tanks_g}}^t & \dots & {Pass^s}_{N_{tanks_g}}^{N_{times}} \end{pmatrix}$$

For the matrix elements, the condition is met: $pass_g^{st} \in \{0, 1\}$. Next, for the genetic algorithm, the matrix is represented as a vector. To do this, first, each matrix is pulled into a vector. Then, the obtained vectors are concatenated.

3 The algorithm of scheduling

In production, the scheduling process can take several days. Since there may be changes in the calendar task, as well as changes in the state of the equipment, the optimal schedule should be planned as quickly as possible. At the same time, it is necessary to calculate not only the optimal schedule, made with fixed terms of products' certification but also the possible terms of accelerated preparation of petroleum products. A solution is required that can reduce the resources used by production through moving the period, lasting from the start of the production to the product certification, to an earlier date. The time of the petroleum product manufacturing should be also reduced, if possible. The optimal schedule should reflect the most productive process of manufacturing petroleum products.

Let us consider an approach to problem-solving, based on the use of a genetic algorithm.

Step 1. Initialization of the initial population. Each individual is a binary vector corresponding to a multidimensional matrix $Pass$ that characterizes the beginning and end of the certification for commercial tanks.

Step 2. The initial values of matrix F^k are generated, where $k = 0$.

Step 3. $k = k + 1$ is taken. The optimization problem is solved for an individual of the population $Pass^k$. In this case, solution F^k is required that satisfies the constraints described earlier in the mathematical model [Zyk2020, Sav2019, Zyk019]. To assess the optimality of the solution, the values of the quality and mass matrices should be calculated.

Step 4. If there is a solution to the optimization problem, then the fitness of the individual $Pass^k$ is evaluated. Otherwise, the transition to step 3 occurs.

Step 5. Selection, crossover, and mutation operations are carried out.

Step 6. The condition for stopping the genetic algorithm is checked. If the condition is met, then one individual or a set of individuals with the best fitness scores for all generations is taken as the answer. Otherwise, the transition to step 3 occurs.

4 Solution method

To solve the optimization problem, the *SciPy* library with the optimize package, namely, the *minimize* method is used. The value of the parameter $method = trust - constr$ sets the algorithm selected for solving the optimization problem. The $trust - constr$ method is a trust region algorithm for solving optimization problems with constraints. It switches between the two implementations depending on the definition of the problem. This is the most universal minimization algorithm with constraints implemented in *SciPy*, and the most suitable for large-scale problems. For problems with equality constraints, it is the implementation of the $Byrd - OmojokunTrust - Region\ SQP$ method. When inequality constraints are also set, the trust-constr method switches to the interior point method. The interior point algorithm, in its turn, solves the problem with inequality constraints by introducing stock variables and solving a sequence of problems by the method of barrier functions with equality constraints for gradually decreasing values of the barrier parameter. The sequential programming method (SQP) with equality constraints is used to solve subtasks with an increase in the level of accuracy as the iteration approaches the solution [Byr99].

However, the direct application of the sequential quadratic programming method to the barrier method leads to inefficient primary steps that, as a rule, violate the positivity of weak variables, and, thus, are often interrupted by a restriction of the trust region. The formulation of the sequential quadratic programming iteration and the definition of the scaled trust region allow us to avoid too close an approximation to the boundary of the admissible domain.

5 Search for an admissible point

The barrier method takes an admissible point as an initial one. In the optimal scheduling problem, selecting an initial point for the optimization algorithm in an admissible domain is a timeconsuming task. The following scheme for searching for an initial admissible point is proposed.

The algorithm for finding the initial admissible point (y_0, x^0) for the problem

$$\begin{gathered} y \to min \\ v_i(x) \le y,\ i = 1, ..., m, \\ h_j(x) = 0,\ j = m+1, ..., n. \end{gathered} \tag{1}$$

is as follows.

Step 1. Initial point x^0 is obtained that satisfies the equality constraints $h_j(x) = 0,\ j = (m+1), ..., n$.

Step 2. $y_0 = max\{v_i(x^0) \mid i = 1, ..., m\}$ is taken.

Step 3. The condition for stopping the algorithm is checked. If $y_0 < 0$, then $v_i(x) \le 0,\ i = 1, ..., m$, and the initial admissible point (y_0, x^0) for problem (1) is found.

Otherwise, starting from the initial point (y_0, x^0), the following minimization problem with equality constraints is solved.

$$\begin{gathered} y \to min \\ v_i(x) - y = 0,\ i = 1, ..., m, \\ h_j(x) = 0,\ j = m+1, ..., n. \end{gathered}$$

until we get $y < 0$.

When the solution $y < 0$ is found, then $v_i(x^*) < 0,\ i = 1, m$, therefore, an admissible point for problem (1) is found.

6 Computational experiment

To test the proposed solution of the optimization problem, a computational experiment was conducted. To do this, consider the process flow chart of the petroleum products manufacturing through blending components. This flow chart should reflect the basic logic of the sequence of actions taken during the commercial production of petroleum products (Figure 1).

The chart shows the main objects needed for modelling the commercial production of petroleum products. Each object has the values of its inherent characteristics. The description of the objects is given in Table 1.

The following is the description of the implemented solution. The first step is to solve the problem of finding an admissible point. The initial point is constructed to be passed to scipy.optimize.root function. The initial

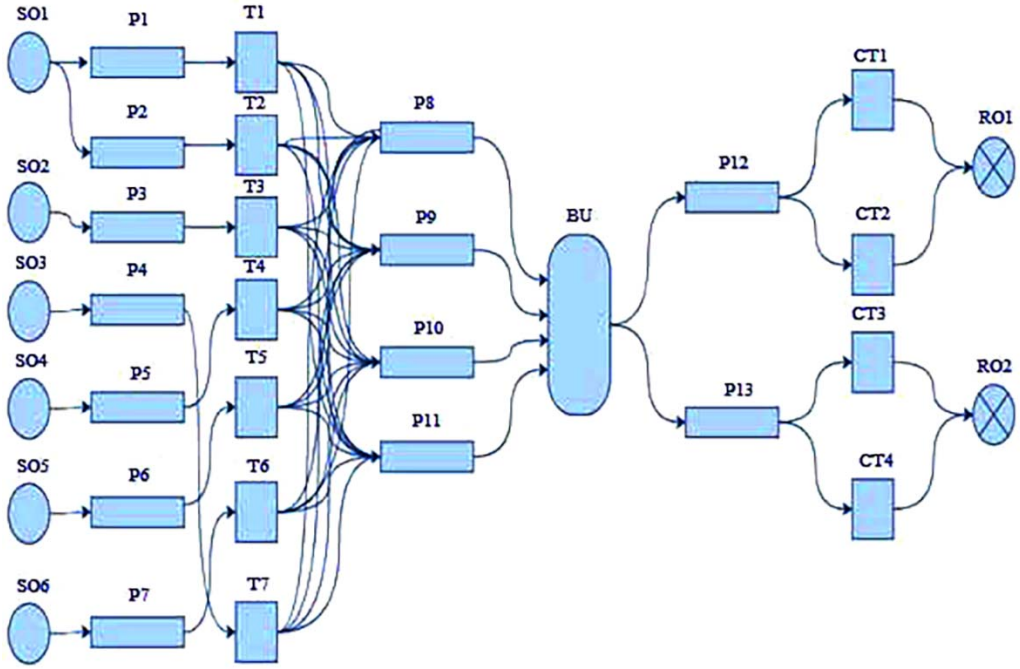

Figure 1: Process flow chart

Table 1: Type of the equipment of the commercial production model

Nomenclature of the flow chart	Type of equipment
SO1, SO2, SO3, SO4, SO5, SO6	Source object
P1, P2, P3, P4, P5, P6, P7, P8, P10, P11, P12, P13	Piping
T1, T2, T3, T4, T5, T6, T7	Tank
BU	Components' blending unit
CT1, CT2, CT3, CT4	Commercial tank
RO1, RO2	Receiving object

point is matrix F of flow rates along the connections of objects at each moment with the dimension of $(56, 12)$ filled with zeros. The result of the *scipy.optimize.root* function is a point that satisfies a system of nonlinear equations consisting of a set of equality constraints.

The next step is the *scipy.optimize.minimize* function, which is used in the following form:

```
  fromscipy.optimize import minimize res = minimize(fun_y, y_f_ vec.ravel(),
method=trust-constr, hess=lambda x:  np.zeros((len(y_f_vec), len(y f vec))),
constraints= [linear_constraint, nonlinear_ constraint], bounds=bounds)
```

Vector f *vec* is obtained by pulling matrix F into a vector. Vector y f *vec* is obtained by adding variable y to vector f *vec*. The constraints of the problem of finding an admissible point are similar to those of the original optimization problem. Consider the imposed boundary constraints on the first value of vector y f *vec*, i.e. variable y.

To end the search for an admissible point, variable y must become a negative value during the optimization process. Even though the values y should lie within the boundaries $(-\infty; 0)$, *scipy.optimize.minimize* works better with specified boundaries $(-1; 0)$. In this case, the search for a solution stops when the value ofy becomes close to -1.

The difficulty of restrictions application lies in the way they are set. The *scipy.optimize.minimize* function

with the $method = trust - constr$ parameter accepts $LinearConstraint$ objects as linear constraints. In this object, the constraints are set by multiplication AF, where F is the velocity matrix of the form (n), and matrix Ahas the form (m, n). To create a LinearConstraint object, it is necessary to express F from the formulae contraints and then create matrix A.

As a result, the *scipy.optimize.minimize* function returns an admissible point for solving the main optimization problem. The found admissible point is taken to a function of the following type:

```
  res_op = minimize(fuction_F, f_vec.ravel(), method=trust-constr,
constraints= [linear_ constraint, nonlinear_constraint], bounds=bounds)
```

As a result of the function, we get a solution to the optimization problem. The solution satisfies the constraints specified in the mathematical model. The value of the target function at the optimal point is 1.525 E-12.

The operating time (Table 2) depends on the parameters of the *scipy.optimize. minimize* function that are taken into account. The function uses the values of the Hessian matrix to find an admissible point. For the considered constraints, the specified matrix is filled with zeros. During the experiment, the Hessian matrix is transferred to the *scipy.optimize.minimize* function of an additional optimization problem. When transferring the main optimization problem to the function, no improvement in the optimal point construction time was found in this experiment. In this case, the Hessian matrix is passed as a parameter when creating constraint objects for the main and additional optimization problems.

Table 2: Operating time of the *scipy.optimize.minimize* function

The problem to be solved	Operating time
Search for an admissible point	11 minutes 55 seconds
Search for an optimal point	15 minutes 36 seconds

The solution needed for creating an optimal schedule can be obtained with the method used, combining the barrier methods, trust region strategies and sequential quadratic programming, with correctly selected parameters and a pre-calculated admissible initial point. The $trust - constr$ algorithm is effective because it avoids too close an approximation to the boundary of the admissible region in search of the optimal solution. The time to develop a solution is conditioned by the set of constraints for the specified process flow chart, as well as by the presence of non-linear constraints, i. e. restrictions on the sequence of filling and emptying of commercial tanks.

7 Conclusion

A computational experiment confirmed the effectiveness of using the *scipy.opti− mize.minimize* function for the problem of optimal scheduling of commercial production of petroleum products. The following further work with this function is required: analysis of quality formulae for the problem of optimal scheduling of petroleum products; implementation of the constraints transformed into the required form due to the selected quality formula; and subsequent implementation of the algorithm for optimal scheduling proposed in the first section using the obtained solution.

References

[Hus2021] I. A. Huseynov, E. A. Melikov, N. A. Khanbutaeva, I. R. Efendiyev. Models and algorithms of a multi-level control system for installations of primary oil refining. *Izvestiya RAS. Theory and control systems*, 1:83–91, 2012.

[Zyk2018] A. V. Zykina, M. Yu. Savelev, T. Yu.Fink. Multi-level management of oil refining production. Requirements for research tasks. *Omsk Scientific Bulletin*, 162(6): 271-274, 2018.

[Zyk018] A. V. Zykina, M. Yu. Savelev, T. Yu.Fink. Characteristics of the dispatch control process in multi-level management system of oil refining. *Applied Mathematics and Fundamental Computer Science*, 5(2): 4-13, 2018.

[Sav2018] V. N. Savkin, A. M. Motkov, M. Y. Saveliev. Comparison of approaches to the calculation of quality indicators of petrols. *Newsletter of the Omsk Scientific and Educational Center of OmSTU and IM SB RAS in the field of mathematics and computer science*, 2(1): 94–97, 2018.

[Zyk2019] A. V. Zykina, O. N. Kaneva, M. Yu. Savelev, T. Yu. Fink. Automation issues in multi-level control systems of petroleum refinery. *AIP Conference Proceedings*, pp. 050015, 2019.

[Zyk2020] A. V. Zykina, O. N. Kaneva, V. N. Savkin, T. Yu. Fink. Problem statement for preparing a single batch of end product under uncertainty. *Lecture Notes in Computer Science*, Volume 11974: 519–527, 2020.

[Sav2019] V. N. Savkin, A. V. Zykina. Mathematical description of the problem of preparing one party of commodity products. *Newsletter of the Omsk Scientific and Educational Center of OmSTU and IM SB RAS in the field of mathematics and computer science*, 3(1): 133–136, 2019.

[Zyk019] A. V. Zykina, O. N. Kaneva, V. N. Savkin, T. Yu. Fink. Calculation methods for quality indicators of commercial gasolines (petrochemicals). *AIP Conference Proceedings*, pp. 020030, 2019.

[Byr99] Richard H Byrd [et al.]. An interior point algorithm for large-scale nonlinear programming. *ISIAM Journal on Optimization*, 9(4): 877–900, 1999.