Federal State Institution of Higher Professional Education
Saint-Petersburg State University
Graduate School of Management

FUEL DELIVERY NETWORK OPTIMIZATION FOR GAZPROM NEFT COMPANY

Master's Thesis by the 2nd year students
BM.5783.2021

PANCHENKO Anastasiia

SEMENOVA Anna

SIVASH Elizaveta

Research adviser:
Ph.D. of Physical and Mathematical Sciences,
Associate Professor of the Department of
Informational Technologies in Management

STRAKHOVICH Elvira Vitautasovna

Reviewer:
Chief Specialist in the Department of
Performance Analysis of
Fuel-Conducting network,
LTD Gazprom Neft Regional Sales

MARKOV Danil Vadimovich

Saint-Petersburg

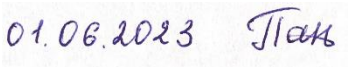
2023

ЗАЯВЛЕНИЕ О САМОСТОЯТЕЛЬНОМ ВЫПОЛНЕНИИ ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЫ

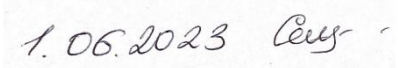
Мы, Панченко Анастасия, студентка 2 курса магистратуры ВМ.5783.2021 «Бизнес-аналитика и большие данные (Master in Business Analytics and Big Data – MiBA)», Семенова Анна, студентка 2 курса магистратуры ВМ.5783.2021 «Бизнес-аналитика и большие данные (Master in Business Analytics and Big Data – MiBA)», и Сиваш Елизавета, студентка 2 курса магистратуры ВМ.5783.2021 «Бизнес-аналитика и большие данные (Master in Business Analytics and Big Data – MiBA)», подтверждаем, что в нашей магистерской диссертации на тему «Оптимизация сети доставки топлива для компании Газпром Нефть», представленной в службу обеспечения программ магистратуры для последующей передачи в государственную аттестационную комиссию для публичной защиты, не содержится элементов плагиата.

Все прямые заимствования из печатных и электронных источников, а также из защищенных ранее курсовых и выпускных квалификационных работ, кандидатских и докторских диссертаций имеют соответствующие ссылки.

Нам известно содержание п. 9.7.1 Правил обучения по основным образовательным программам высшего и среднего профессионального образования в СПбГУ о том, что «ВКР выполняется индивидуально каждым студентом под руководством назначенного ему научного руководителя», и п. 51 Устава федерального государственного бюджетного образовательного учреждения высшего профессионального образования «Санкт-Петербургский государственный университет» о том, что «студент подлежит отчислению из Санкт-Петербургского университета за представление курсовой или выпускной квалификационной работы, выполненной другим лицом (лицами)».



Панченко Анастасия



Семенова Анна

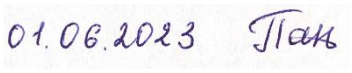
Сиваш Елизавета

STATEMENT ABOUT THE INDEPENDENT CHARACTER OF THE MASTER THESIS

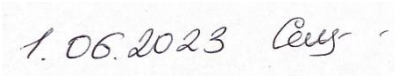
We, Panchenko Anastasiia, 2nd year master student of the BM.5783.2021 master program «Master in Business Analytics and Big Data – MiBA», Semenova Anna, 2nd year master student of the BM.5783.2021 master program «Master in Business Analytics and Big Data – MiBA», and Sivash Elizaveta, 2nd year master student of the BM.5783.2021 master program «Business Analytics and Big Data (Master in Business Analytics and Big Data - MiBA)», state that our master thesis on the topic «Fuel Delivery Network Optimisation for Gazprom Neft Company» which is presented to the Master Office to be submitted to the Official Defense Committee for the public defense, does not contain any elements of plagiarism.

All direct borrowings from printed and electronic sources, as well as from master theses, PhD and doctorate theses which were defended earlier, have appropriate references.

We are aware that according to paragraph 9.7.1. of Guidelines for instruction in major curriculum programs of higher and secondary professional education at St. Petersburg University «A master thesis must be completed by each of the degree candidates individually under the supervision of his or her advisor», and according to paragraph 51 of Charter of the Federal State Institution of Higher Professional Education Saint-Petersburg State University «a student can be expelled from St. Petersburg University for submitting of the course or graduation qualification work developed by other person (persons)».



Panchenko Anastasiia



Semenova Anna

Sivash Elizaveta

TABLE OF CONTENTS

INTRODUCTION

In recent years, the topic of effective resources distribution and transportation organization in particular has attracted attention of many corporations and researchers alike. There are more than 10 thousand research papers written and published according to EBSCO on the topics of resource allocation and the transportation problem. A well-built logistics model helps to lower transportation costs, which is especially relevant if these constitute a large proportion of a company's overall expenses. The latter is true for an oil and gas industry, where operations include fuel delivery on a regular basis.

This thesis paper addresses a fuel delivery network optimization for Gazprom Neft company and involves a development of an automated model of fuel transportations from oil bases to petrol stations. The scope of this work is on the level of FLD (Fuel & Logistics Department), which has a mission of guaranteeing minimal cost of oil products at salespoint by timely fulfilling the needs to the full extent, while preserving the product quality. Therefore, the business problem itself is focused on minimizing operational expenses (OPEX) on logistics. To be more specific, the fuel delivery network includes pairing of oil bases and petrol stations and the related volumes needed to be delivered from first to second. For each pair there are concrete costs calculated based on tariffs; these depend on the distance, volumes, and type of the product. Thus, the model was supposed to match oil bases and petrol stations, so that the overall OPEX is minimal.

Although this task could be performed manually, the business representatives were interested in creating a model which would automate this process and save time and labor costs. At the same time, there were several restrictions that should be taken into account. Firstly, there was concrete demand for each brand at each petrol station that was to be fully satisfied. Secondly, oil bases had a certain volume of each product available that could not be exceeded. Thirdly, there was a so-called «one to one» limitation, which meant that each petrol station must get all brands from one and only one oil base. Finally, management could set a loading threshold for an oil base, so that its supply was used up to a certain degree. As an expected result the model optimally distributes volumes of each product between oil bases and petrol stations, considering all limitations and ideally providing a result in 25 minutes or less; the latter was the requirement set by the business representatives.

Prior to creating a model, a literature review was carried out to specify the main groups of methods related to the solution of a transportation problem; almost all of them refer to linear programming. The methods differ in the initial mathematical meaning, optimization approach, complexity and time needed for calculation of result. Then, based on the business request, the mathematical model

was formulated. After that, there was a need to understand how a concrete mathematical model dictates the type of a problem to be solved; the important parameters to consider were a target function, limitations and variable data types. Finally, specific tools (such as python libraries, packages, solvers) were chosen to technically solve the specific problem. Overall, the following *research goal* was formulated: to determine the most suitable methods and tools given existing business requirements. The *business goal* of the thesis was to minimize transportation costs (OPEX) for delivery network of Gazprom Neft company.

Since this paper is a research-orientated thesis, the *goal of the thesis* was to determine the most suitable methods and tools given existing business requirements for minimization of transportation costs for delivery network of Gazprom Neft company. In order to achieve it, the following steps were taken:

1. Investigate and analyze existing methods for solving similar problems and estimate their applicability to the described case. This task was done by conducting sufficient literature review on linear programming and adjacent areas;
2. Describe the mathematical form of the model based on business requirements. This part means selection of needed parameters and their unification in several mathematical equations that would describe the main minimization function and restrictions;
3. Choose suitable methods based on literature review and on the mathematical model and make technical realization(s) of the model. As was stated, the result of this project was an automated model, thus it was decided to apply programming language (Python) and associated libraries and packages for its creation. There are different tools available in Python to solve similar problem, so there was a need to compare them;
4. Compare resulting models (built based on different tools) and select the best approach in terms of managerial criteria;
5. Visualize results in order to represent obtained improvement in the logistics model. Here the focus was on the solution itself and on the financial effect;
6. Provide recommendations for user interactions with model and error handling.

The research paper also attempted to answer the following question: «Which of the existing methods in the linear programming area and related approaches is mostly applicable to the fuel delivery transportation problem in terms of current business restrictions?»

As for the research design, it was applied research dedicated to the transportation problem solution on the company level of analysis. As was mentioned before, the business goal of the study

was to minimize operational costs on logistics, and there are plenty of approaches that one can rely on to achieve this goal; however, the definition of the optimization problem and the related choice of concrete methods and tools remain ambiguous. Therefore, the rationale of research is related to the development a systematic approach that describes connections between various optimization problems, methods, and tools. The philosophy of positivism is related to this; this philosophy is based on the construction of mathematical models to develop an objective approach. Here, theory is of analytic type, the research design was exploratory, because solvers and libraries are being investigated, so that then, based on a set of criteria, the best ones were chosen. Then the research approach of reasoning was mixed. It was partly inductive because at first, there was data, from which an understanding was formed what task should be solved, a theory was built. But approach was also deductive because then there was a movement from understanding the theory and the place of the current problem in it to solving a specific problem based on the data provided.

The data for the project was provided by Gazprom Neft in a user-friendly format (tabular format in csv and excel), so that no additional data collection and extraction were needed. Some minor adjustments were made to transform data into more convenient for the model format. The data contained several parts: firstly, pairs of oil bases and petrol stations with distances, tariffs, and products available for transportation, regions, and months of transportations; secondly, the demand of each petrol station in terms of volumes of brands; thirdly, the capacity of each oil base in terms of volumes of products; finally, the current distribution of products between oil bases and petrol stations as a baseline.

The research type is based on quantitative methods. The quantitative methods included secondary data analysis and identification of important parameters for the model, formulation of the mathematical model and its technical realization with the usage of Python programming language. In particular, the latter included data preparation, model building, and comparison of the results yielded by different technical realizations of the model. To visualize and present the results to business parties, Python programming language was used.

As for the structure of the thesis paper, the first chapter focuses on the overview of the theoretical background of the transportation problem, its place in the linear programming and adjacent areas, and a variety of mathematical methods and technical tools that can be used to solve the problem. In the second chapter, the data is described, a mathematical model is formulated, and several technical realizations of the model are created. In the final chapter, findings of the research are presented; man-

agerial criteria are applied to compare different versions of the model; and discussion on user interaction with model and error handling is provided. Finally, in conclusion, the importance of the conducted research is provided, as well as its practical and theoretical implications and limitations.

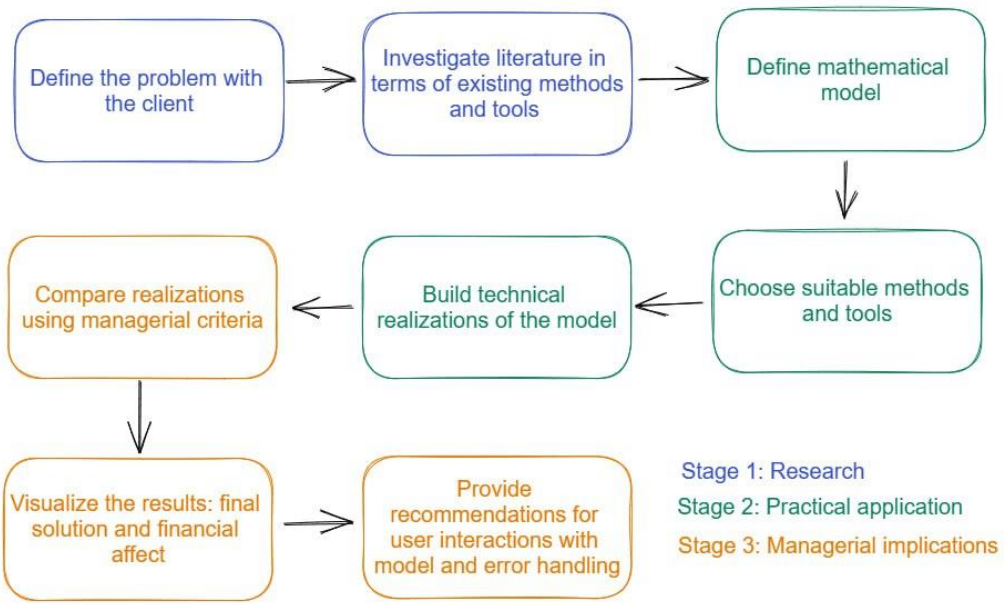


Figure 1. Project plan. Source: (Provided by authors, 2023)

The sequence of actions during the project including research, practical application and managerial implication is presented in Figure 1 above.

CHAPTER 1. OVERVIEW OF APPROACHES TO TRANSPORTATION PROBLEM

1.1. Transportation problem and its importance

As was mentioned in the previous section, the stated research problem refers to the linear programming area, to be more specific, to the transportation problem. The transportation problem tries to identify the best possible distribution of products needed to be delivered. It is aimed at minimizing shipping costs by optimally allocating products between origins and delivery points (Reeb & Leavengood, 2002). Basically, the transportation problem is about finding an optimal balance between supply and demand on each side, considering additional constraints, if any.

Initially the theory of resource allocation was created and formalized by Monge (1781), as an optimal transportation problem. Further, it was also developed and advanced by Kantorovich (1942), which transformed it into the Monge-Kantorovich transportation problem. The idea of applying this theory to creation of minimization cost function when delivering goods from one point to another refers to Hitchcock-Koopmans transportation problem, and the paper is based mainly on this formulation of the problem (Ford & Fulkerson, 1956). Information on historic development is summarized in Figure 2.

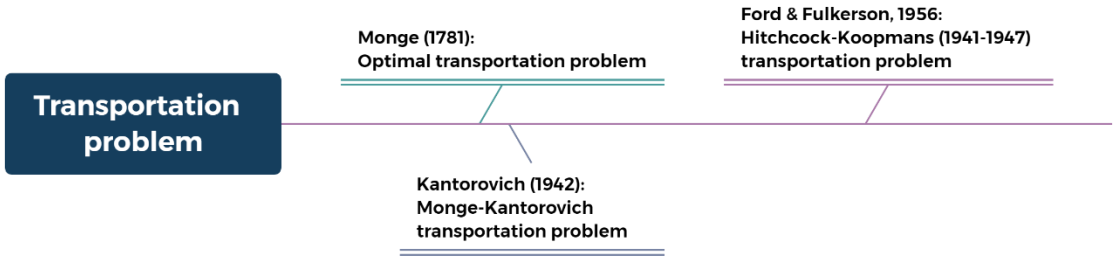


Figure 2. Historic development of transportation problem. Source: (Provided by authors, 2023)

In order to solve the transportation problem in general terms, the following information is needed: number of origins, number of points (or destinations), the quantity or volume of available product on each origin (supply), the quantity or volume of needed product on each point (demand), the unit cost of shipping the product from each origin to each point. In case of equal demand and supply, the transportation problem is called balanced; in another case, when demand and supply are not equal the transportation problem is unbalanced. Unbalanced transportation problem is represented in Table 1.

Table 1. Unbalanced transportation problem. Source: (Ranasinghe, 2021)

Sources	Destinations						Supply
	D_1	D_2	....	D_j	....	D_n	
S_1	X_{11} C_{11}	X_{12} C_{12}	....	X_{1j} C_{1j}	.....	X_{1n} C_{1n}	a_1
S_2	X_{21} C_{21}	X_{22} C_{22}	....	X_{2j} C_{2j}	.....	X_{2n} C_{2n}	a_2
. .	. .	. .	....	. .	.....	. .	. .
S_i	X_{i1} C_{i1}	X_{i2} C_{i2}	....	X_{ij} C_{ij}	.....	X_{in} C_{in}	a_i
. .	. .	. .	....	. .	.....	. .	. .
S_m	X_{m1} C_{m1}	X_{m2} C_{m2}	....	X_{mj} C_{mj}	.....	X_{mn} C_{mn}	a_m
Demand	b_1	b_2	....	b_j	....	b_n	$\sum_{i=1}^{m} a_i > \sum_{j=1}^{n} b_j$ or $\sum_{i=1}^{m} a_i < \sum_{j=1}^{n} b_j$

1.2. Review of optimization problems

So, the transportation problem about finding an optimal balance between supply and demand on each side, considering additional constraints, is to be solved. However, before solving it, it is necessary to understand what place it occupies in the whole variety of optimization problems. The potential approaches for optimization depend on many factors — the function that is to be optimized, the constraints that should be satisfied, the type of variables that are required for these constraints, and so on. Based on this knowledge, it will be possible to understand which solver and which library or package in Python is suitable for the practical solution of the transportation problem.

A deeper dive into the topic of optimization problems should be started with the fact that optimization in mathematics is such a task, the result of which should be finding the extreme value (minimum or maximum) of the objective function, subject to certain restrictions (Levitin & Polyak, 1966). Mathematical programming studies the theory and numerical methods of optimization. Optimization is used not only in mathematics, but also in a number of other disciplines, for example, economics and engineering. In addition, machine learning, which has received great development now, is also based on optimization methods (Gambella et al., 2021).

Optimization can be constrained and unconstrained depending on the availability of constraints, where the former is considered if the conditions specified in a set of linear or nonlinear equalities or inequalities must be met. The latter, unconstrained optimization, is the one where there is only

a function that needs to be optimized, without a set of certain restrictions (Bertsekas, 1982). For this study, the problem of conditional optimization is relevant, since there is a certain set of business requirements in the form of restrictions on the demand of petrol stations, the supply of oil bases and others, which will be discussed in more detail in the practical part.

Further, optimization is divided into local and global. The global one looks for the actual minimum or maximum of the function over the entire area, and the local one searches for one of the many extremes which is not necessarily global (Hiriart-Urruty & Lemaréchal, 2004). Here it is necessary to move on to the concept of convex and nonconvex programming. An optimization problem is convex if the objective function is convex, as well as the domain of feasible solutions. Any local solution of a convex problem is also global, so they do not differ, so there is only one optimal solution. A Non-Convex Optimization (NCO) problem is one where the objective function or any of the constraints is nonconvex. The solution obtained as a result of such optimization may be local, but not global, and therefore nonconvex optimization is considered more complex (Birolini et al., 2021). *Linear programming (LP)*, which refers to the problem studied in this study and involves linear objective function and linear constraints, refers to convex programming, so it is much easier to solve than nonlinear programming (NLP) problems, which can often relate to nonconvex optimization (Diwekar, 2020).

Convex optimization also includes convex quadratic programming (QP) with linear constraints or with convex quadratic constraints. This QP, which is a problem of optimizing a quadratic function, belongs to NLP. Like linear programming, such quadratic programming is a special case of second-order cone programming or SOCP. SOCP refers to semi-defined programming or SDP (Bazaraa, 2013), because SOCP constraints are possible in representation as linear matrix inequalities (LMI). At the same time, SDP is a particular case of cone programming and can be solved, like SOCP, by the method of internal points, which will be discussed below. A more detailed consideration of these concepts is beyond the scope of this paper.

To get a more complete picture of the existing optimization problems' context, it is also necessary to mention optimization problems with power cone constraints (POW) and problems with exponential constraints (EXP). The first type of problem is related to the case when using power cones, it is possible to express constraints (MOSEK, n.d.). The second type of problems introduces the concept of the exponential cone, which is applied to models with both exponential and logarithmic constraints. However, these two types of optimization problems also lie outside the scope of current study.

Moving on, the variables that are searched for in the optimization problem can be set not only in float form, but also in the form of integer values. Hence, linear and nonlinear programming tasks

can also be set, for example, as integer linear programming (ILP) and integer nonlinear programming (INLP), if variables can only be integer. There are mixed integer linear programming (MILP) and mixed integer nonlinear programming (MINLP), if part of the variables is float, and the part is integer (Lee & Leyffer, 2011). A special case of integer programming is the situation when variables are binary, that is, they can be 0 or 1.

In general, the variety of optimization problems can be represented as in Figure 3.

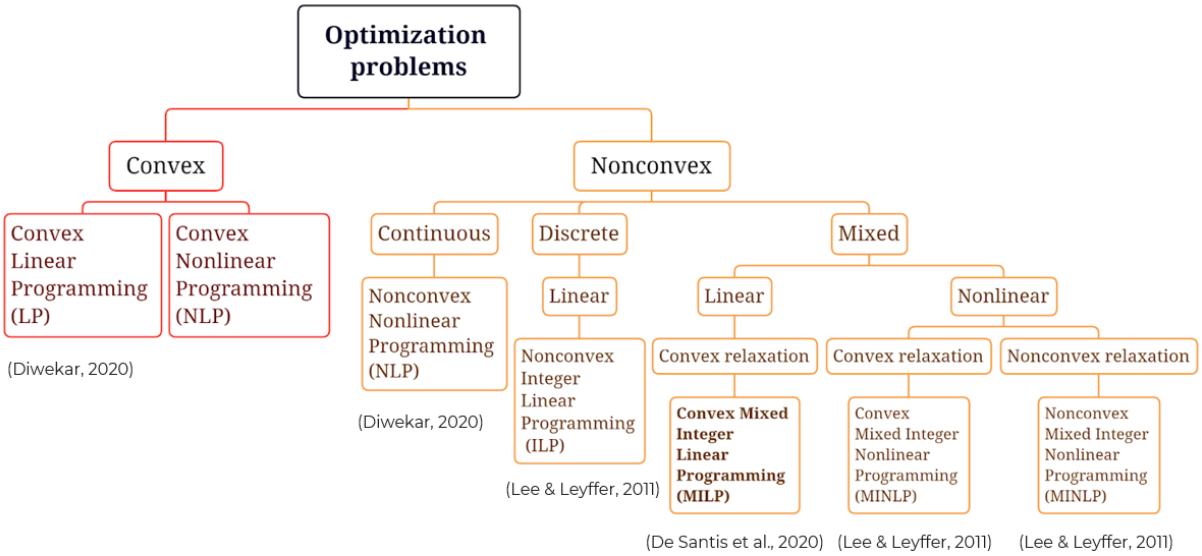


Figure 3. Taxonomy of optimization problems. Source: (Provided by authors, 2023)

Figure 3 is a taxonomy of optimization problems; key parameters to consider are types of variables, whether the problem is convex or nonconvex, as well as if objective function and constraints are linear or nonlinear.

In this study, according to the taxonomy above, the mixed integer linear programming (MILP) problem is to be solved, since one of the constraints requires the introduction of a binary variable taking the value 0 or 1 and other variables are float. At the same time, the minimization function remains linear, and all constraints are described by linear dependencies, which means the problem does not go beyond linear programming. This conclusion about the location of the current transportation problem in taxonomy was made during a practical study, which is to be described in more detail in the second chapter.

The concept of relaxation is presented in the taxonomy for mixed integer problems. The fact is that the mixed integer programming problem is an NP-hard problem that is computationally difficult to solve. The presence of discrete variables leads to structural nonlinear programming, that is, to the

nonconvexity of the problem. Therefore, a relaxation method is used, which is standard for approximating complex optimization problems, in which a related problem is to be solved in polynomial time. For example, variables from 0 to 1 are considered, instead of taking the restrictions strictly 0 or 1. Due to this relaxation, the problem becomes convex programming with linear constraints and continuous variables, then an optimistic assessment of the solution of the program is obtained, that is, it is possible to approximately solve the original problem on its basis (De Santis et al., 2020).

Due to the fact that the problem in this study relates to MILP, the task of finding that solver and, accordingly, finding a library or package in Python that would be suitable for this particular case appears. Such libraries, packages and solvers are to be discussed further in this chapter.

1.3 Methods of solving transportation problem

After mathematical and logical formulation of the transportation problem is completed, the two steps are usually taken — firstly, a basic feasible solution is found, and secondly, this solution is optimized.

1.3.1. Basic feasible methods

As for the basic feasible solution, there are several methods for it, such as North-West Corner Rule, Least Cost Method, and Vogel Approximation method.

The North-West Corner Rule

One of the earliest methods is the North-west corner rule developed by Charnes et al. (1953). It takes as an input the supplies and demands with costs in a matrix form and firstly checks the problem for balance. If it is unbalanced, there is a need to introduce a dummy variable for the remaining demand and to balance the problem. After that by availability and demand the method starts resource allocation from the north-west cell in the matrix (or top left cell), checks the smallest possible value for supply and demand for this cell, and checks the difference in supply and demand. Then, it removes the corresponding matrix row or column and repeats the procedure until all resource allocations are done (Ranasinghe & Rathnayaka, 2021). So, during this method a step-by-step procedure is made, thus making it simple to obtain an optimal solution.

The Least Cost Method

The second mentioned method is the Least cost method, also called minimum matrix method, which was formulated at the very beginning of Hitchcock transportation problem development and is now considered as more accurate than North-west corner rule (Hossain, et al., 2020). In this method

the procedure includes allocation of as many deliveries as possible with the smallest unit cost cell (Uddin et al., 2016).

At the first stage of the least cost method, similar to the north-west corner rule, the problem should be balanced. Secondly, the smallest unit cost cell should be found and the minimum of supply and demand at this point should be highlighted. Then the cells with lower costs are selected instead of cells with higher cost in order to provide less total cost for transportation. As a result, the optimal allocation strategy is more accurate and effective than in the north-west method because the delivery costs are taken into account in the process of allocation, and in the beginning.

The Vogel Approximation Model

Finally, the most accurate out of listed three methods is Vogel approximation method, suggesting the idea of so-called «penalties» as difference between current minimal and following costs in the matrix rows and columns. When the highest «penalty» is found the cell with corresponding lowest costs is highlighted (Hakim, 2012). Due to such a system, the method itself is more complex than the previous two described, however provides better solutions (Uddin et al., 2016).

1.3.2. Optimizing methods

As for the optimization of the basic solution, the following methods can be used: Simplex method, Stepping-stone method, Method of potentials, and others.

The Stepping-stone method

This method includes an algorithm for finding the potential of non-basic variables in terms of the objective function (Charnes & Cooper, 1953). It is applied after finding a basic feasible solution by one of the methods (VAM, north-west corner, or least cost), and basically checks if it is optimal or not. It determines the effect on the transportation cost in case one unit is assigned to the empty cell (Wulandari & Arifin, et al., 2018).

The Simplex method

The simplex method for optimization is considered a relatively effective and computationally compact method for finding minimum/maximum, which was developed by Dantzig in 1947 (Nelder & Mead, 1965). The simplex method is effective because it does not take each point of the function for calculation but moves from corner to corner of a convex polyhedron in the feasibility region and optimizes the value of objective function (Dantzig, 1956). So, the calculations begin with the «starting» basic solution, and then a search is conducted for solutions that would improve the value of the

objective function. This is possible only if an increase in some variable leads to an increase in the value of the functional.

The method of potentials

The second mentioned method, method of potential (Gass, 2003), is an advanced version of simplex method, which also starts from some feasible solution and tries to find the most optimal one. This method is a modification of the simplex in a way that the base matrix is represented as a tree. Determination of the output arc and recalculation of potentials is implemented quite effectively. The main «consumer» of time is the optimality check, however there are different ways to speed up calculations.

Depending on what optimization problem is to be solved, a certain solver is used, inside of which an algorithm is laid down that is suitable for this task. For example, if there is a linear programming problem with continuous variables, then usually either the simplex method discussed above or the Interior-point method is implemented inside solvers.

The Interior-point method

The basic idea of the variant of the Interior-point method for linear programming is to replace the constraints with a penalty using a barrier function, and the property of the function guarantees that there is a minimum for the desired variable (Karmarkar, 1984). First, an initial approximation is selected within the allowable range, and then the minimum for each iteration is determined using the Newton approximation method, depending on the set parameter, monotonically decreasing to zero. All together minimums form a central path and give an approximate solution to the original problem.

If the transportation task is mixed integer programming, then the Branch and bound method and the Cutting-plane method are applied.

The Branch and bound method

The Branch and bound method is essentially a variation of a complete search of acceptable solutions with the elimination of subsets of solutions that do not contain optimal (Land & Doig, 1960). It is based on the divide and conquer approach, and the task is divided into many subtasks. The area is divided into subdomains, as a result forming a search tree. In other words, there are a lot of branching based on rules, each time there is a dropout, and the current optimal solution is selected. At the end, when the stop rule is triggered, that is, there are no active subsets left, the optimal solution is selected, which at the moment was the best.

The Cutting-plane method

The Cutting-plane method is an optimization method that refines a valid set or objective function iteratively using linear inequalities, which are called cuts. The result is a sequence of relaxations that limit the solution space, and as a result, a solution for the original integer problem is found (Gomory, 1958). This method is used not only to obtain solutions to MILP problems, but also to solve problems that do not necessarily relate to differentiable convex optimization problems.

The Branch and Cut method

The Branch and Cut method, which, for example, is implemented in the CBC solver (Forrest & Lougee-Heimer, 2005) for some Python libraries, combines the Cutting-plane method and the Branch and bound method described above (Mitchell, 2002). This is applied in such a way that in any subtask, the Cutting-plane method is performed as long as there are cutting planes. After this, the transition is performed by one of the remaining fractional variables.

The methods discussed are summarized in Figure 4. It can be seen that some of them refer to combinatorial methods, while others to continuous. The first means such types of methods which refer to the problem where some of the variables are set to be discrete (Schrijver, 2003). In turn, combinatorial methods can be divided into approximate and exact algorithms, where sub-optimal algorithms with provable guarantees about the quality of the solutions are called approximation algorithms (Trevisan, 2011). However, this work is focused only on exact algorithms, thus, the approximation ones are not considered in detail. The second branch of optimization methods, continuous, one oppositely refers to cases when variables in the objective function are required to be continuous (Jeyakumar & Rubinov, 2006).

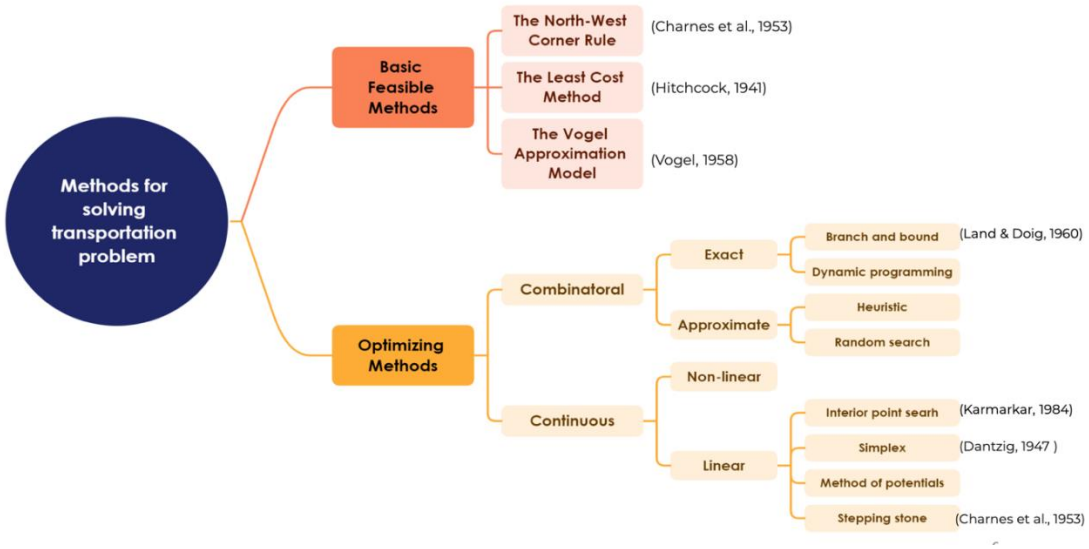


Figure 4. Taxonomy of optimization methods. Source: (Provided by authors, 2023)

1.4. Review of existing Python libraries

Python provides access to numerous packages and libraries that can be used for solving transportation problems. Here only some of these will be listed along with a specification for which kind of problem in particular they can be applied to and whether they are open source or not. Most of the information on solvers is stated on the respective packages' websites.

PuLP (PuLP, 2022) is a linear-programming modeler in Python; it allows access to the following solvers:

- GLPK — free; problems: LP.
- CBC — free; problems: LP, MILP.
- CPLEX — commercial; problems: LP, MILP, MILNP (QP, SOCP), NLP (QP, SOCP).
- GUROBI — commercial; problems: LP, MILP, MINLP (QP, SOCP), NLP (QP, SOCP).
- MOSEK — commercial; problems: LP, MILP, MINLP (QP, SOCP, EXP, POW; except SDP), NLP (QP, SOCP, SDP, EXP, POW).
- XPRESS — commercial; problems: LP, MILP, MINLP (QP, SOCP), NLP (QP, SOCP).
- CHOCO — free; problems: CP.
- MIPCL — free; problems: MILP.
- SCIP — free; problems: LP, MILP, MINLP (QP, SOCP), NLP (QP, SOCP).

SciPy (SciPy, n.d.) is a Python library that allows to perform scientific and engineering calculations, among other things, it can be used to solve a transportation problem. SciPy grants access to the following solver:

- COBYLA — free; problems: NLP.
- SLSQP — free; problems: NLP.
- trust-constr — free; problems: NLP.
- HIGHS — free; problems: NLP, LP, MILP, (not MINLP!).

COBYLA, SLSQP, trust-consrt are available from scipy.optimize.minimize package, HIGHS — via scipy.optimize.milp package.

CVXPY (CVXPY, n.d.) is a free modeling language for finding solutions to convex optimization problems; CVXPY is realized in Python (Diamond & Boyd, 2016). It allows access to multiple solvers; among those you can find:

- Already mentioned above: CBC, GLPK, CPLEX, GUROBI, MOSEK, SCIP, XPRESS.
- CLARABEL — free; problems: LP, NLP (QP, SOCP, EXP, POW).
- COPT — free; problems: LP, NLP (QP, SOCP, SDP), MILP (not MINLP!).
- GLOP — free; problems: LP.
- GLPK_MI — free; problems: LP, MILP.
- OSQP — free; problems: LP, NLP (QP).
- PROXQP — free; problems: LP, NLP (QP).
- PDLP — free; problems: LP.
- ECOS_BB — free; problems: LP, NLP (QP, SOCP, EXP), MILP, not MILNP.
- CVXOPT — free; problems: LP, NLP (QP, SOCP, SDP).
- SDPA — free; problems: LP, NLP (SDP).
- SCS — free; problems: LP, NLP (QP, SOCP, SDP, EXP, POW).

Pyomo (Pyomo, n.d.) is a free optimization modeling language with various optimization capabilities available in Python (Hart et al., 2017). Some of the solvers that Pyomo supports include:

- Already mentioned above: CBC, GLPK, CPLEX, MOSEK.
- Bonmin — free; problems: LP, MILP, NLP, MINLP.
- IPOPT — free; problems: NLP, LP
- CONOPT — free; problems: NLP.
- Couenne — free; problems: MINLP.

- FilMINT — free; problems: MINLP.
- Filter — commercial; problems: NLP (QP).
- KNITRO — commercial; problems: MILP.
- OOQP — free; problems: LP, NLP (QP).

CVXOPT (CVXOPT, n.d.) is an open-source software package for convex optimization based on Python. Among the solvers that CVXOPT supports are:

- Already mentioned GLPK, MOSEK.

It should be noted that this list cannot be considered a full list of solvers/libraries, there are other solutions as well, both free and commercial.

The information on packages, problems and solvers is summarized in Figure 5.

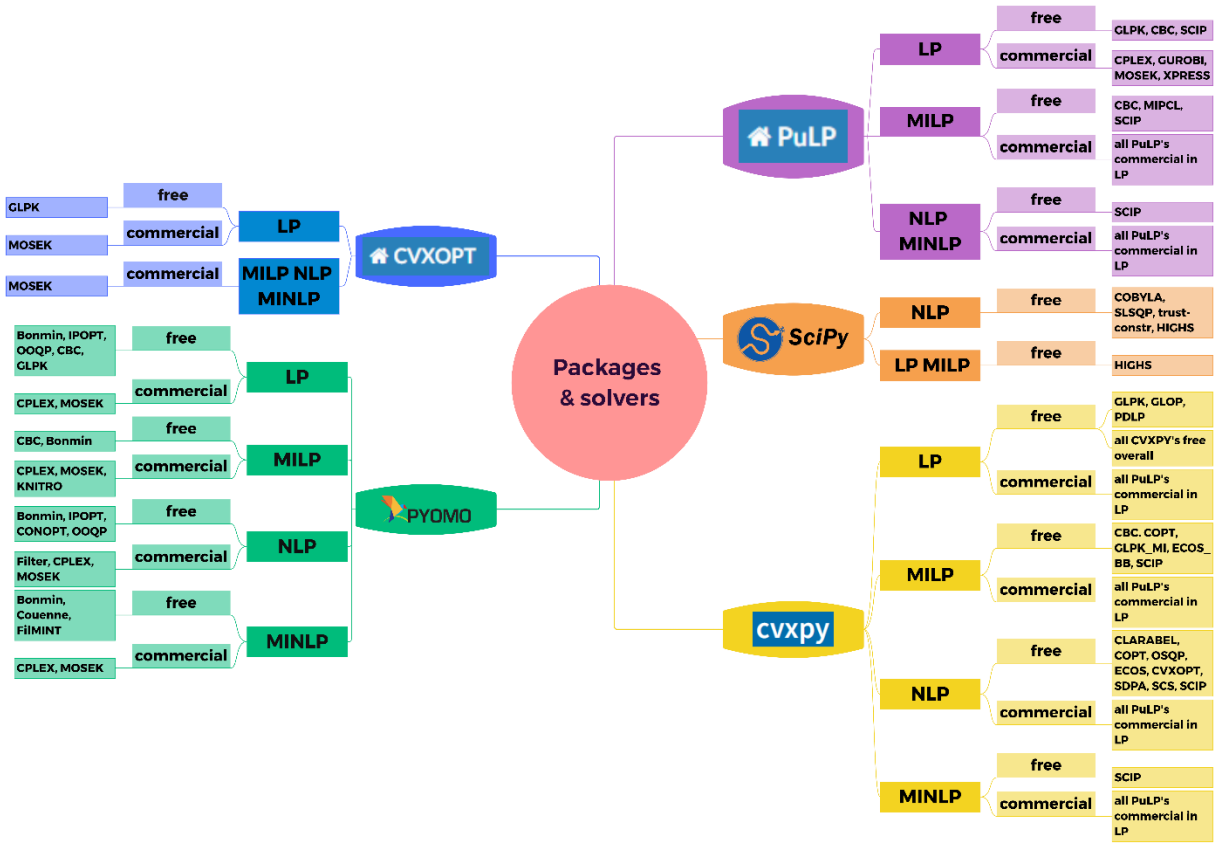


Figure 5. Mind map of packages and solvers. Source: (Provided by authors, 2023)

1.5. Summary of chapter 1

In this chapter the definition of a transportation problem was covered, as well as its importance and historic development.

In addition, a larger topic of optimization problems was described — the area to which a transportation problem logically belongs; here the focus was on the definition of a convex and non-convex

optimization, linear and nonlinear objective function and restrictions, continuous and discrete data types. This information is summarized in Figure 3 that is a taxonomy of optimization problems, such as LP, NLP, MILP, and MINLP.

Furthermore, a list of methods that can be used to solve a transportation problem was provided, including two main groups of methods: basic feasible solution methods (North-West Corner Rule, Least Cost Method, and Vogel Approximation method) and optimizing methods (Simplex method, Stepping-stone method, Method of potentials, and etc.). In Figure 4 information on these methods has been summarized in a form of taxonomy, most important parameters to consider were whether this is a combinatorial or continuous optimization, an exact or approximate algorithm, a linear or nonlinear problem.

Finally, some of the Python libraries and packages that can be used to solve a transportation problem were described, such as SciPy, PuLP, Pyomo and others. This information is summarized in a mind map in Figure 5 that specifies to which solvers these libraries/packages allow access to, which problems these solvers can deal with, and whether solvers are commercial or open-source.

Logical connections between a transportation problem, optimization problems, methods and tools are summarized in Figure 6 which is a concept map.

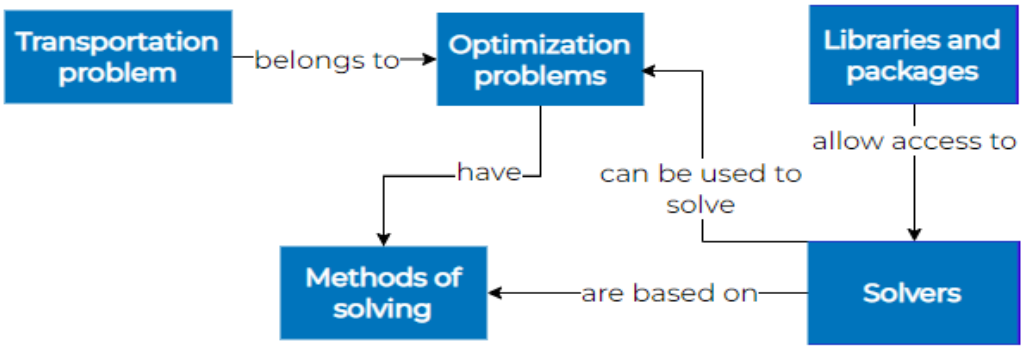


Figure 6. Concept map of Chapter 1. Source: (Provided by authors, 2023)

To sum up, the transportation problem belongs to a larger field of optimization problems. Various optimization problems have different methods of solution; in Python it is possible to find libraries and packages that allow access to numerous solvers; these solvers are based on methods and can be used to solve optimization problems.

CHAPTER 2. METHODOLOGY FOR FUEL DELIVERY NETWORK OPTIMIZATION

2.1. Data description

The data for this project was collected and provided by Gazprom Neft company, therefore the data was secondary. Initially, there were a test dataset, which was used for the model creation, and a respective baseline dataset to compare model's solution with; this version of the data only contained information for one region and for two months. Later, the company provided an extended dataset for multiple regions and for twelve months to test the model with and a new baseline solution dataset, both files were presented in excel format. Description provided below is relevant for these latest versions of datasets.

The first file which is the main one contained all the necessary information about parameters such as the bandwidth, or supply, of oil bases, the demand of various oil brands that must be met for petrol stations, information about types of oil products, logistic legs between oil bases and petrol stations, as well as data on several kinds of tariffs for logistics. Information about the tariffs was parsed by representatives from Gazprom Neft company from the internet. Each petrol station and oil base were uniquely located in regions; a petrol station from one region could get the products from an oil base from another region if a connection between them was specified in the data. The second file contained a baseline solution compiled by the company for each month, with which model's solution could be compared in terms of overall costs for transportation.

Overall, as the solution the model determines the optimal pairs of oil bases and petrol stations, as well as delivery volumes of products to petrol stations and overall costs of their transportation. A more detailed description of the data is presented in Table 2.

Table 2. Description of data parameters. Source: (Provided by authors, 2023)

Parameter	Description	Type of data	Measurement
Date	The month for which the information is specified and for which resources need to be allocated (all months of 2023, twelve in total)	Categorical	-
Region	The region for which the information is specified (16 regions in total)	Categorical	-
Oil base	The name of the oil base from where the product can be delivered; each name corresponds to a unique index. There are 43 oil bases in total.	Categorical	-

Parameter	Description	Type of data	Measurement
Petrol station	Places where oil products should be delivered. Each petrol station has a unique index. There are 1280 petrol stations in total.	Categorical	-
Brand	The type of oil product to be distributed. Currently there are 18 groups in data. For example, there are such brands as gasoline 100 brand, gasoline 92, gasoline 95, gasoline 95 brand, summer diesel fuel with additives.	Categorical	-
Product	Short name of petroleum products excluding information about brands for gasoline. These petroleum products are DF, PE100, PE92, PE95, PE98. Multiple brands can be produced from the same product.	Categorical	-
Transport fleet	From which transport fleet fuel trucks are used. It can be third-party and own. For this project, a third-party one is used, but the own one can be also used when scaling the project.	Categorical	-
Logistic leg	Distance from the oil base to the petrol station	Numerical	Kilometers
Delivery area	The category to which the logistic leg belongs. There are, for example, the following categories: up to 50 km, from 51 to 150, from 151 to 300 and more than 300 km. Costs of secondary logistics depend on this area.	Categorical	-
Railway tariff	The cost of transporting products by railway to oil bases. The tariff varies depending on the month, as well as different for gasoline and diesel fuel.	Numerical	Rubles per ton
Storage tariff	The cost of storing petroleum products in cisterns at an oil base. The tariff depends on the specific oil base.	Numerical	Rubles per ton
Brand tariff	The cost of the branding is accounted because special additives are added to gasoline or diesel that improve their quality. The tariff depends on the brand.	Numerical	Rubles per ton

Parameter	Description	Type of data	Measurement
Secondary logistics tariff	Costs for secondary logistics from oil bases to petrol stations. If the delivery area belongs to a category which is more than 50 km of distance, then the tariff for secondary logistics is multiplied with the logistics leg, if less, then a fixed tariff for secondary logistics is taken. This tariff depends on the distance itself and on the transport fleet which is described above in the table.	Numerical	Rubles per ton
Tariff overall	Overall tariff calculated specified for a concrete match of petrol station, oil base and brand. If no information on tariffs is given, it is considered that there is no connection for a given match.	Numerical	Rubles per ton
Volume of demand	Demand at each petrol station by types of petroleum brands	Numerical	Tons
Volume of supply	Supply of each oil base by types of petroleum products	Numerical	Tons
Baseline	Baseline total transportation costs; the details of the baseline were the following: real and planned costs were given for a concrete match of oil base, petrol station and brand.	Numerical	Rubles

2.2. Data exploration

In this section some aggregated information on the data is given. First, match between products and brands is described, second, regions are analyzed in terms of supply and demand, third, petrol stations and oil bases are examined in more detail. It should be noted that if in the current month for a trio of <oil base-petrol station-brand> the information on tariffs was given, but no matching <oil base-product> or <petrol station-brand> were found in supply and demand (or if found values were equal to zero), such connections were deleted.

As was mentioned before, on the side of the demand, petrol stations specify the volumes they need in brands; on the side of the supply, oil bases list the volumes they can potentially provide in

products. The match between products and brands is described in Table 3 for the situation of this exact project. Later this particularity in the data introduces some complications in the mathematical model.

Table 3. Match between products and brands. Source: (Provided by authors, 2023)

Product	Brand
PE100	Gasoline 100 brand
	Gasoline 100 brand with additives
PE95	Gasoline 95
	Gasoline 95 brand
	Gasoline 95 optimum
PE92	Gasoline 92
	Gasoline 92 optimum
DF	Summer diesel fuel with additives
	Summer diesel fuel brand
	Summer diesel fuel
	Winter diesel fuel with additives class 2
	Winter diesel fuel brand class 2
	Winter diesel fuel class 2
	Diesel fuel with additives demi-season sort F
	Diesel fuel demi-season brand sort F
	Diesel fuel demi-season sort F
	Diesel fuel Artic class 4
	Diesel fuel Artic with additives class 4

It could be noticed that no brand related to the product PE98 was mentioned in Table 3. However, in one variation of mathematical model this product can be used to produce other gasolines.

Although the model is intended to be used on the whole list of regions at the same time, it makes sense to analyze the ratio of supply and demand on the regional level. In Table 4 there is analysis of 16 regions in terms of supply and demand; there is information about numbers of petrol stations and oil bases that belong to each region. It is also specified for each region for how many products (out of four possible with exception PE98 for which no demand is given) supply exceeds demand, and for how many — demand exceeds supply. Then, using that information, the percent of products where supply exceeds demand was calculated by division of products where supply exceeds demand by overall number of products. It can be seen that some regions can be considered as donor «donor-regions» and some — «recipient-regions».

Table 4. Analysis of regions' supply and demand for January 2023. Source: (Provided by authors, 2023)

Region	Number of petrol stations	Number of oil bases	Supply > Demand	Demand > Supply	Supply > Demand, %
Region 1	137	1	2	2	0.50
Region 2	445	3	1	3	0.25
Region 3	395	3	0	3	0.00
Region 4	146	5	3	1	0.75
Region 5	51	1	2	1	0.67
Region 6	325	3	1	3	0.25
Region 7	405	3	0	4	0.00
Region 8	570	5	0	4	0.00
Region 9	325	1	0	4	0.00
Region 10	583	4	0	4	0.00
Region 11	469	2	0	4	0.00
Region 12	469	3	0	4	0.00
Region 13	446	1	0	4	0.00
Region 14	484	1	0	3	0.00
Region 15	370	5	0	4	0.00
Region 16	370	2	1	3	0.25

In Table 5 below for all the products it is checked whether supply exceeds demand. Such check can be built for any set of regions; in the current version it is presented for all regions.

Table 5. Analysis of products for January 2023. Source: (Provided by authors, 2023)

Product	Demand	Supply	Supply > Demand
PE100	7 392	27 191	True
PE92	199 039	410 165	True
PE95	232 510	357 547	True
DF	305 101	586 426	True
PE98	0	363	True

In Figure 7 below the information on distribution of supply volumes between oil bases is summarized. It can be seen that these volumes are not equally distributed. It could lead later to some problems with satisfaction of lower bound of supply restriction, which is discussed further in details.

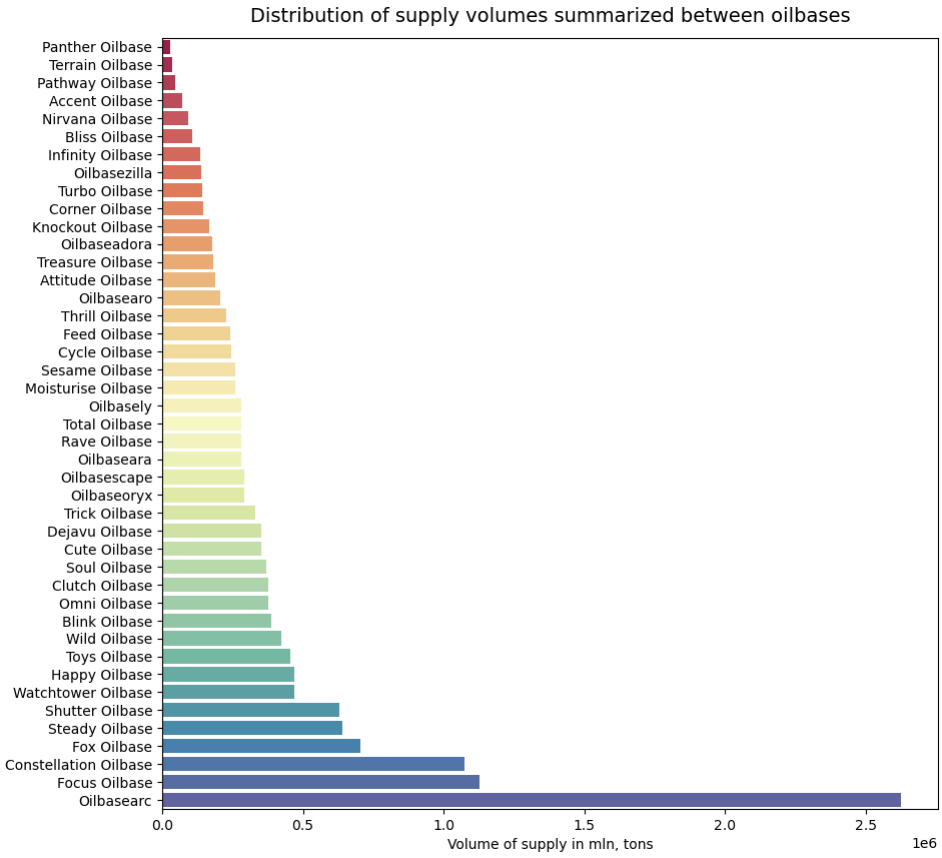


Figure 7. Distribution of supply volumes between oil bases for January 2023. Source: (Provided by authors, 2023)

In Figure 8 below the boxplot for summarized demand on petrol stations is given. Although one might expect for demand to be evenly distributed among petrol stations, it seems clear from the boxplot that some petrol stations have a surprisingly high value for a demand.

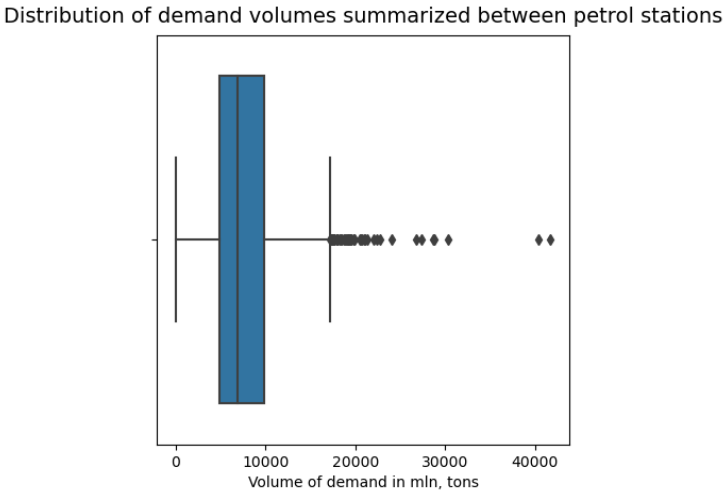


Figure 8. Distribution of demand volumes between petrol stations for January 2023. Source: (Provided by authors, 2023)

Table 6. Supply and related demand, January 2023. Source: (Provided by authors, 2023)

Origin	Supply	Related petrol stations	Related demand	Supply / Related demand
Focus Oilbase	95 825	137	72 646	1.32
Rave Oilbase	23 957	26	18 454	1.30
Watchtower Oilbase	39 926	51	31 714	1.26
Blink Oilbase	33 005	32	27 527	1.20
Sesame Oilbase	21 959	26	18 454	1.19
Soul Oilbase	31 082	32	27 527	1.13
Oilbasearc	192 811	445	286 576	0.67
Constellation Oilbase	89 611	287	144 588	0.62
Happy Oilbase	39 926	120	79 915	0.50
Fox Oilbase	59 897	325	177 346	0.34
Feed Oilbase	20 677	120	79 915	0.26
Steady Oilbase	54 847	469	261 417	0.21
Infinity Oilbase	55 897	445	286 576	0.20
Omni Oilbase	31 940	325	177 346	0.18
Wild Oilbase	35 935	370	202 658	0.18
Oilbasezilla	31 941	370	202 658	0.16
Thrill Oilbase	30 413	370	202 658	0.15
Oilbaseadora	24 048	265	173 150	0.14
Dejavu Oilbase	29 944	405	217 400	0.14
Oilbaseara	23 957	325	177 346	0.14
Bliss Oilbase	23 956	325	177 346	0.14
Shutter Oilbase	39 355	565	307 773	0.13
Total Oilbase	23 956	370	202 658	0.12
Accent Oilbase	23 896	370	202 658	0.12
Clutch Oilbase	31 785	484	275 320	0.12
Trick Oilbase	27 949	446	243 954	0.11
Cute Oilbase	29 945	469	261 417	0.11
Oilbasely	23 956	405	217 400	0.11
Oilbaseoryx	24 812	469	261 417	0.09
Panther Oilbase	1 630	26	18 454	0.09
Oilbasescape	24 812	565	307 773	0.08
Knockout Oilbase	14 036	325	177 346	0.08
Attitude Oilbase	16 032	405	217 400	0.07
Cycle Oilbase	20 700	583	315 236	0.07
Pathway Oilbase	18 609	565	307 773	0.06
Moisturise Oilbase	15 434	395	258 394	0.06
Treasure Oilbase	15 468	469	261417	0.06
Turbo Oilbase	11 978	370	202 658	0.06
Oilbasearo	11 943	393	257 629	0.05
Nirvana Oilbase	7 600	583	315 236	0.02
Terrain Oilbase	6 243	565	307 773	0.02

In Table 6 above the information of supply and related demand for all oil bases is described. Firstly, for each oil base overall possible supply is specified. Then, petrol stations related to these supplies are considered and sums of related demands are represented for these petrol stations. Finally, the proportions of supplies to related demands for each oil bases were calculated and presented in table in order from the highest to the smallest values. It should be noted that for the month specified there are some unusual cases where the proportions of supplies to related demands exceed 1. This may mean that it is worth expanding the network of petrol stations so that the demand would correspond to the existing supply.

2.3. Mathematical model

This section is organized as follows: firstly, to make the explanation of the following formulas simpler, a description of all the multidimensional arrays that are used in this paper is provided, along with all the necessary notations. Secondly, at first, costs' calculation is briefly covered; thirdly, objective function and all the restrictions are defined; finally, full mathematical model is provided.

2.3.1. Usage of multidimensional arrays

Firstly, there is a **Demand** matrix of two dimensions (or 2D tensor), the matrix is present in formula 1. It contains K rows and N columns. Here K is equal to the number of brands in the data and N is equal to the number of petrol stations.

$$Demand = \begin{vmatrix} P_1\_brand_1 & \dots & P_N\_brand_1 \\ \vdots & \ddots & \vdots \\ P_1\_brand_K & \dots & P_N\_brand_K \end{vmatrix}_{K \times N} \quad (1)$$

For example, $P_1\_brand_1$ is a constant that takes nonnegative continuous values; if $P_1\_brand_1$ is equal to zero, it means that at petrol station 1 there is no demand for brand 1.

Secondly, there is a respective **Supply** matrix of two dimensions (or 2D tensor), the matrix is present in formula 2. The only major difference is that while for demand the details are provided in brands, in supply the details are in products. The same product can be used to produce various brands. Supply matrix contains L rows and M columns. Here L is equal to the number of products in the data and M is equal to the number of oil stations.

$$Supply = \begin{vmatrix} O_1\_product_1 & \dots & O_M\_product_1 \\ \vdots & \ddots & \vdots \\ O_1\_product_L & \dots & O_M\_product_L \end{vmatrix}_{L \times M} \quad (2)$$

For example, $O_1\_product_1$ is a constant that takes nonnegative continuous values; if $O_1\_product_1$ is equal to zero, it means that at oil station 1 there is no supply for product 1.

Apart from Demand and Supply matrices, there is also a **Volume** multidimensional array; Volume is a matrix of 3 dimensions (or 3D tensor), which is present in formula 3. Volume is a tensor of depth K, N rows and M columns. Here K is equal to the number of brands, N is the number of petrol stations, M is the number of oil bases.

$$Volume = \left\| \begin{vmatrix} P_1O_1\_brand_1 & \dots & P_1O_M\_brand_1 \\ \vdots & \ddots & \vdots \\ P_NO_1\_brand_1 & \dots & P_NO_M\_brand_1 \end{vmatrix}, \dots, \begin{vmatrix} P_1O_1\_brand_K & \dots & P_1O_M\_brand_K \\ \vdots & \ddots & \vdots \\ P_NO_1\_brand_K & \dots & P_NO_M\_brand_K \end{vmatrix} \right\|_{K\times N\times M} \quad (3)$$

Here $P_1O_1\_brand_1$ is a variable that takes nonnegative continuous values; after the problem is solved, it can take the value of zero in two cases. First, if there is a demand for brand 1 at the petrol station 1, but another oil base was chosen. Second, if there is no demand for brand 1 at the petrol station 1 and (or) no supply for brand 1 at the oil base 1, it is to be again set to zero by the model because of the constraints.

Next, there is a **Cost** multidimensional array. Cost is also a matrix of 3 dimensions (or 3D tensor), its description is provided in formula 4. Its shape is the same as the shape of Volume, the difference is that its elements take constant continuous non-negative values; some of its elements take the value of value B, which stands for a conditionally infinite constant, or a big number **B** ($B = 10^9$); the latter is true if transportation is impossible by the conditions of the task.

$$Cost = \left\| \begin{vmatrix} Cost\_P_1O_1\_brand_1 & \dots & Cost\_P_1O_M\_brand_1 \\ \vdots & \ddots & \vdots \\ Cost\_P_NO_1\_brand_1 & \dots & Cost\_P_NO_M\_brand_1 \end{vmatrix}, \dots, \begin{vmatrix} Cost\_P_1O_1\_brand_K & \dots & Cost\_P_1O_M\_brand_K \\ \vdots & \ddots & \vdots \\ Cost\_P_NO_1\_brand_K & \dots & Cost\_P_NO_M\_brand_K \end{vmatrix} \right\|_{K\times N\times M} \quad (4)$$

There is also a need to focus on separate components that are later used for costs' calculations. There is vector **R** that is a railway tariff, its values depend on a brand; vector **S** that is a storage tariff, its values depend on an oil base; vector **BR** that is a brand tariff, its values depend on a brand; vector **SL** that is a secondary logistics tariff, its values depend on three parameters: an oil base, a petrol station, and a transportation fleet used. All of these four vectors have constant continuous nonnegative values.

Apart from these, there is also a vector **Distance**, values of which depend on a distance between a specific pair of an oil base and a petrol station and take constant continues nonnegative values, and a vector **Dummy,** the elements of which take the value of one if the respective distance exceeds 50 km, and zero otherwise.

Here is an important note: though each vector can theoretically be of a different length, depending on the parameters it is related to (brand, oil base, petrol station, etc.), for the simplicity of overall computations all the vectors have the same length that is equal to the product of the numbers of brands, petrol stations, and oil bases: $K \times N \times M$; therefore, there are to be some duplications in values of these vectors. The elements inside the vectors should be sorted in such a way that it would be possible to reshape them in the same form as Cost, to form matrices of 3 dimensions (or 3D tensors).

Finally, there is a **Oil_base_choice** matrix of 2 dimensions (or 2D tensor) with N rows and M columns. It consists of binary variables; its description is provided in formula 5.

$$Oil\_base\_choice = \begin{vmatrix} y\_P_1O_1 & \dots & y\_P_1O_M \\ \vdots & \ddots & \vdots \\ y\_P_NO_1 & \dots & y\_P_NO_M \end{vmatrix}_{N \times M} \quad (5)$$

Again, here K is equal to the number of brands, N is the number of petrol stations, M is the number of oil bases.

The last important point to mention is axes; this information is used later for explanation of formulas. Based on the dimensions of a tensor, the number of axes varies. If it is a 1D tensor, there is only one axis (either rows or columns). If it is a 2D tensor, there are two axes: axis 1 is rows, axis 2 is columns. If it is a 3D tensor, there are three axes: axis 1 is depth, axis 2 is rows, and axis 3 is columns.

2.3.2. Costs calculation

In this section, at first, costs' calculation for the model is briefly covered. For the simplicity of overall comprehension, details of costs component calculation are provided separately from the objective function. Costs' calculation is provided below in a matrix form in formula 6, after all the component vectors were reshaped in a matrix with three dimensions, so that the shape of each has the depth of K, N rows, and M columns.

$$Cost = (R + S + BR + SL \odot (1 - Dummy) + SL \odot Distance \odot Dummy) \quad (6)$$

Here S is a storage tariff, BR is a brand tariff, SL is a secondary logistics tariff, $Distance$ is the distance between a specific pair of an oil base and a petrol station. $Dummy$ equals one if the distance between oil base m and petrol station n exceeds 50 km, and zero otherwise. More detailed information can be found in Table 2. The costs are calculated in advance and are considered as constants in the mathematical model.

2.3.3. Definition of objective function

The business goal, as was mentioned before, was to minimize operational expenses on logistics; therefore, in the objective function OPEX should be minimized. Logically, overall OPEX can be calculated as the sum of all multiplications of respective volumes and tariffs. Here one important aspect should be highlighted once again: on the side of the demand, petrol stations specify the volumes they need in brands; on the side of the supply, oil bases list the volumes they can potentially provide in products. In our exact situation, the match between products and brands is given in Table 3.

The volumes for our model are matches between a brand, a petrol station and an oil base. The objective function is given in formula 7.

$$Minimize: Z = \sum Volume \odot Cost, for\ n = \overline{1,N}, for\ m = \overline{1,M},\ k = \overline{1,K} \quad (7)$$

Here both $Volume$ and Cost are multidimensional arrays mentioned in the previous paragraphs. For more details, refer to formulas 3 and 4.

2.3.4. Definition of restrictions

The first restriction states that a concrete demand for each brand at each petrol station is to be fully satisfied. In a mathematical form, the first restriction can be formulated as follows in formula 8.

$$\sum_{m=1}^{M} P_n O_{m\_} brand_k = P_{n\_} brand_k,\ for\ n = \overline{1,N},\ k = \overline{1,K} \quad (8)$$

The component on the left comes from the $Volume$ tensor, where each row is summarized (axis 2 of a 3D tensor is used for summarization); the component on the right comes from the Demand matrix, which consists of constant values of demand at a concrete petrol station n for a specific brand k. Here n is the index of a petrol station, m is the index of an oil base, k is the index of brand. Again, some petrol stations may not have a demand for a particular brand, so in that case the constant term is set to zero.

The second restriction specifies that oil bases have a certain volume of each product available that cannot be exceeded. As the first attempt to transform the second restriction into a mathematical form, formula 9 is present below. Note that this expression is to be reformulated later.

$$\sum_{n=1}^{N} P_n O_{m\_} brand_k = O_{m\_} brand_k,\ for\ m = \overline{1,M},\ k = \overline{1,K} \quad (9)$$

This formula is, in fact, the reversed version of formula 8. The component on the left comes from the $Volume$ tensor, where each column is summarized (axis 3 of a 3D tensor is used for summarization); the component on the right comes from the Supply matrix, which consists of constant values of supply for a concrete pair of an oil base m and a brand k. The issue of <brand-product>

pairs is to be solved, more information on the nature of this problem is provided in Table 3. Logically, brands that are produced from the same product go into the same equation in the second restriction, with a constant constraint that is equal to the volume of a product supply available at an oil base. The first attempt to reformulate formula 9 is present in formula 10:

$$\sum_{n=1}^{N} P_n O_{m\_}product_l \leq O_{m\_}product_l,\ for\ \forall\, m \in M,\ l = \overline{1,L},\quad \forall P_n O_{m\_}product_l = \sum_{s=s1}^{ss} P_n O_{m\_}brand_s,\quad \forall s \in S \tag{10}$$

Here l is the index of products in the Supply matrix, s_1 is the set of indexes of brands that can be made from the same product 1, so **S** is the set of sets. Each of its elements contains sets of brands' indexes with overlapping source, or product. So, for example, Gasoline 95 and Gasoline 95 brand belong to the same set since they are both produced from PE 95. Now, formula 10 is once again transformed in formula 11, so that original variables with brands are used in the main expression:

$$\sum_{n=1}^{N} \sum_{s=s_1}^{ss} P_n O_{m\_}brand_s \leq O_{m\_}product_l,\ m = \overline{1,M},\ l = \overline{1,L},\ \forall s \in S \tag{11}$$

The component on the left comes from the $Volume$ tensor, where each column is summarized (axis 3 of a 3D tensor is used for summarization); the component on the right comes from the $Supply$ matrix, which consists of constant values of supply for a concrete pair of an oil base m and a concrete product l. Here n is the index of a petrol station, m is the index of an oil base, l is the index of product, S is the set of sets for brands with overlapping source. Again, some oil bases may not have a supply for a particular product, so for in that case the constant term is set to zero.

Now it is possible to move to the third restriction «one to one» which basically says that each petrol station must get all brands from one and only one oil base. In order to formulate this in a mathematical form, there is a need to use a set of variables that can take discrete values, as well as an already mentioned big number ($B, B = 10^9$). The mathematical form of the third restriction is present in formulas 12 and 13:

$$\sum_{k=1}^{K} P_n O_{m\_}brand_k \leq B \times y\_P_n O_m,\ m = \overline{1,M}, n = \overline{1,N} \tag{12}$$

$$\sum_{m=1}^{M} y\_P_n O_m = 1, n = \overline{1,N} \tag{13}$$

Here $y\_P_n O_m$ are binary variables that signify the final choice of the oil base, which come from Oil_base_choice matrix, described in more detail in formula 5. For example, for petrol station n, $y\_P_n O_m$ will take the value of one if all brands come from an oil base m, and zero if another oil base

is chosen. So, if its value equals zero, the sum of all brands for a concrete pair of a petrol station n for and an oil base m equals zero as well. In formula 12, if the value of $y\_P_nO_m$ is one, the upper bound in formula will take the value of $B, B = 10^9$. The component on the left in formula 12 comes from the $Volume$ tensor, where depth is summarized (axis 1 of a 3D tensor is used for summarization). Formula 13 states that for a concrete petrol station n only one out of all binary variables $y\_P_nO_m$ (each associated with a concrete oil base) can take the value of one. In a situation when there is no such combination of a petrol station n and an oil base m, it does not matter which one of the binary variables takes the value of one.

Finally, the fourth restriction is to be taken into account: management can set a loading threshold for an oil base, so that its supply is used up to a certain degree. Loading threshold, or **percent_loading**, is a constant that can take the values from zero (0%) to one (100%). Logically, the fourth restriction is the opposite of the second restriction; while the second defines the upper bound for supply, the fourth states the lower bound. A set of oil bases, for which a lower bound is specified, is determined by the user and is called **OTC**, or origins to control. It is impossible to control for all the oil bases at the same time since, overall, supply significantly exceeds supply as shown in Table 5. It is also evident in Figure 7 that some oil bases have a very high volume of supply, so for these oil bases it is likely impossible to set a significant lower bound. For these oil bases that belong to OTC lower bound is equal to $percent\_loading \times Supply$. In a mathematical form, the fourth restriction can be present as in a formula 14:

$$\sum_{n=1}^{N} \sum_{s=s_1}^{ss} P_nO_m\_brand_s \geq percent\_loading \times O_m\_product_l, \; m = \overline{1, M}, \; if \; m \in OTC, \; l = \overline{1, L}, \forall s \in S \tag{14}$$

The component on the left comes from the $Volume$ tensor, where each column is summarized (axis 3 of a 3D tensor is used for summarization); the component on the right comes from the $Supply$ matrix, which consists of constant values of supply for a concrete pair of an oil base m and a concrete product l and multiplied by $percent\_loading$. Here n is the index of a petrol station, m is the index of an oil base, l is the index of product, S is the set of sets for brands with overlapping source, or product. Again, some oil bases may not have a supply for a particular product, so for in that case the constant term is set to zero.

2.3.5. Formulation of mathematical model

In the section above the mathematical model is formulated. It is as follows:

Objective function:

$$Minimize: Z = \sum Volume \odot Cost, for\ n = \overline{1,N}, for\ m = \overline{1,M},\ k = \overline{1,K} \quad (15)$$

Subject to constraints:

$$\sum_{m=1}^{M} P_n O_m\_brand_k = P_n\_brand_k,\ for\ n = \overline{1,N},\ k = \overline{1,K} \quad (16)$$

$$\sum_{n=1}^{N} \sum_{s=s_1}^{s_S} P_n O_m\_brand_s \leq O_m\_product_l,\ m = \overline{1,M},\ l = \overline{1,L},\ \forall s \in S \quad (17)$$

$$\sum_{k=1}^{K} P_n O_m\_brand_k \leq B \times y\_P_n O_m,\ m = \overline{1,M}, n = \overline{1,N} \quad (18)$$

$$\sum_{m=1}^{M} y\_P_n O_m = 1, n = \overline{1,N} \quad (19)$$

$$\sum_{n=1}^{N} \sum_{s=s_1}^{s_S} P_n O_m\_brand_s \geq percent\_loading \times O_m\_product_l,\ m = \overline{1,M},\ if\ m \in OTC,\ l = \overline{1,L},\ \forall s \in S \quad (20)$$

Constraints on variables:

$$P_n O_m brand_k \in R, P_n O_m brand_k \geq 0 \quad (21)$$

$$y\_P_n O_m \in (0,1) \quad (22)$$

Where: $P_n O_m brand_k$ are continuous non-negative variables that come from $Volume$ matrix that contains volumes of the brands transported, $y\_P_n O_m$ are dummy variables signifying the choice of an oil base that come from Oil_base_choice matrix, $Cost$ is a matrix with constant non-negative costs associated with a transportation, $P_n\_brand_k$ (for a petrol station n, brand k) come from $Demand$ matrix and $O_m\_product_l$ (for an oil base m, product l), come from $Supply$ matrix. S is the set of sets of brands that share the same product, *OTC* is the set of oil bases to set a lower bound for.

2.3.6. Variations of mathematical model

It should be noted that there are some possible variations of supply constraints. A user is able to choose the version he or she would like to use. Here a brief explanation is provided.

Upper constraint can be specified in two ways: first, a stricter approach, is described above in formula 17. Second approach states that all the products can be divided in two groups: diesel fuel and gasoline; all brands that are produced from the gasoline products (PE100, PE92, PE95) are here considered to be produced from the same product, the same is true for diesel fuel.

$$\sum_{n=1}^{N} \sum_{g=g_1}^{g_G} P_n O_m\_brand_g \leq \sum_{g=g_1}^{g_G} O_m\_product_g,\ m = \overline{1,M},\ if\ \in OTC,\ \forall g \in G \quad (23)$$

The component on the left comes from the $Volume$ tensor, where each column is summarized (axis 3 of a 3D tensor is used for summarization); the component on the right comes from the $Supply$ matrix, which consists of constant values of supply for a concrete pair of an oil base m and a concrete product l. Here n is the index of a petrol station, m is the index of an oil base, G is the set of sets for brands that come from the same group (gasoline or diesel fuel). Again, some oil bases may not have a supply for a particular product, so for in that case the constant term is set to zero.

As for a lower bound, there are two other possible variations to the approach described in formula 20. In the first one in formula 24 the lower bound is set based on the brands belonging to a group of products, same as the one described above in formula 23.

$$\sum_{n=1}^{N}\sum_{g=g_1}^{g_G} P_n O_{m\_brand_g} \geq percent\_loading \times \sum_{g=g_1}^{g_G} O_{m\_product_e},\ m = \overline{1,M},\ if\ \in OTC, \forall g \in G, \tag{24}$$

In the second approach in formula 25 below the overall sum by all products is taken in the right part; in the left part of the equation, all the brands for the same oil base are summarized. This approach is the less strict out of all three, since groups and products are ignored here.

$$\sum_{n=1}^{N}\sum_{k=1}^{K} P_n O_{m\_brand_k} \geq percent\_loading \times \sum_{l=1}^{L} O_{m\_product_l},\ m = \overline{1,M},\ if\ m\ \in OTC \tag{25}$$

2.4. Definition of optimization problem

Previously in Figure 3 classification of optimization problems was provided; now the following Figure 9 is used to identify the exact type of the problem formed by the mathematical model above. The model has binary variables, therefore, by definition the optimization problem is nonconvex.

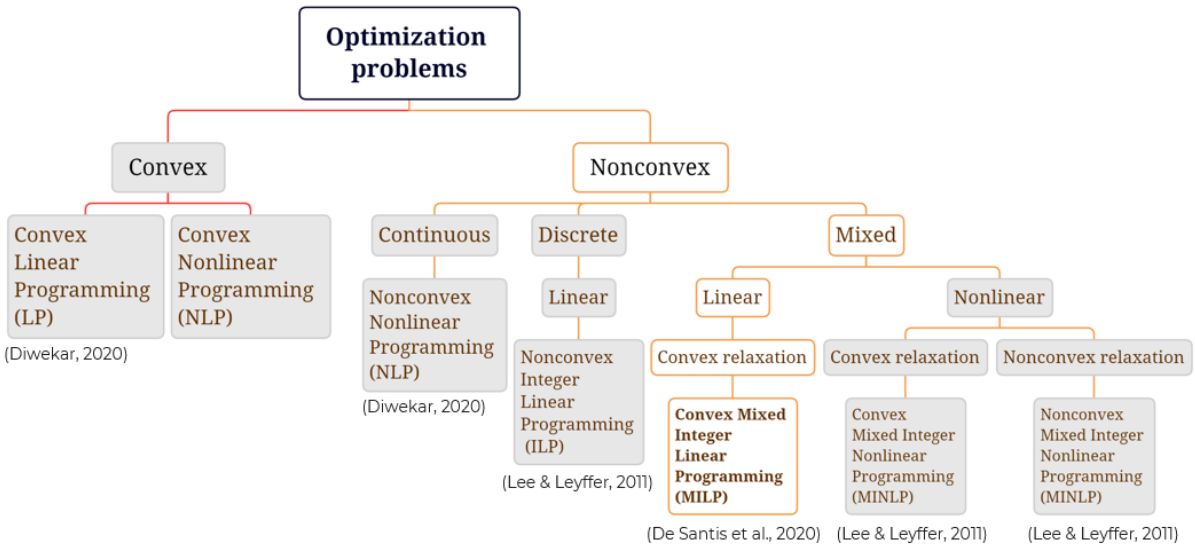


Figure 9. Taxonomy of optimization problems (selected problem type). Source: (Provided by authors, 2023)

At the same time, there are also continuous variables, so overall, the data types are mixed, and, consequently, convex relaxation is applied. Therefore, the problem is Convex Mixed Integer Linear Programming, or MILP. The logic described above is present in Figure 9, which is a modified version of Figure 3. The grey color shows nonrelevant paths, the white demonstrates the relevant one.

2.5. Choice of solvers

Previously in Figure 5 the mind map of packages, libraries and solvers was created. Given that the optimization problem is MILP, the suitable tools are present in Figure 10, again a modified version of an already existing Figure 5.

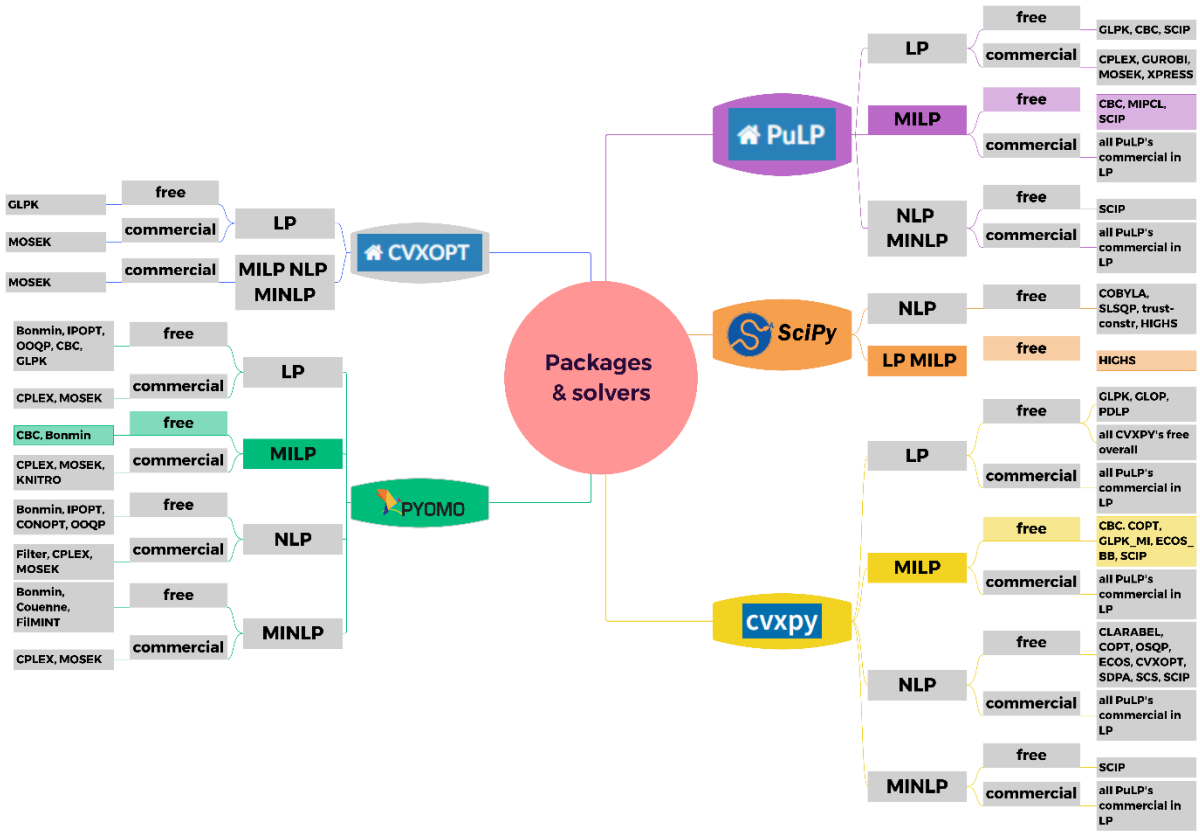


Figure 10. Mind map of packages and solvers (chosen ones). Source: (Provided by authors, 2023)

Here, again the grey color shows the nonrelevant paths, and colored demonstrate the relevant ones. Commercial solvers were marked as nonrelevant in this paper.

2.6. Overview of overall approach

This subsection is dedicated to the description of an overall approach applied to the model building as to the project. To begin with, the agile (incremental and iterative) approach was used, meaning that all constraints were applied gradually and on each iteration the version of model was

improved and became more complex. Different libraries were applied to fulfill constraints. After each iteration the results were discussed with Gazprom Neft representatives.

In the beginning only the first two constraints (petrol station's demand constraint and oil base's supply constraint) were applied, thus the problem was classified as convex linear programming initially. During this step, three libraries were tested — PuLP, SciPy, and CVXOPT. Those libraries suggested opportunities to solve the classical transportation problem. Probably, the most complex and problematic library out of these three was SciPy, as it worked only with matrix form of data which increased the probability of making mistakes when arranging the matrixes for all products.

However, after adding the third and fourth constraints it became clear that the type of model that was chosen in the beginning was incorrect, therefore adjustments were implemented. Firstly, the problem was re-identified as convex mixed integer linear, because a binary variable was introduced in the third constraint (concerning delivering all products from the same oil base). Based on this, the choice of libraries also changed, because some of the mentioned (for instance, CVXOPT) did not support this type of problem. The PuLP was the only one out of three which supported the MILP problem. So, new libraries were tested, — Pyomo and CVXPY, which contain free solvers to deal with MILP. After applying all four needed constraints and testing different libraries and solvers, three results to compare were obtained. In order to choose the best, it was needed to apply several managerial criteria, and to consult with Gazprom Neft colleagues.

2.7. Technical implementation

In this subsection the description of each library and package used is provided. There will be a note on difference between them according to several criteria, such as solvers/methods available, main data structures, time of model running (depending on solvers), and general commentary on the libraries' syntax, flexibility, and readability. It should be mentioned that not all libraries that were discussed previously are described in this subsection, as some of them were not suitable for all constraints.

For example, the SciPy library turned to be rather complex in terms of data structure, as it required matrix form. For the simple linear programming problem with only one type of product there would be probably more sense to use it, however in our case it increased the chance of manual mistake and would be complicated for further automation. Consequently, it was decided not to describe this library further. Similarly, the CVXOPT library was also not included in this technical description due to absence of support for mixed integer problems. Though there were attempts to apply CVXOPT on first iterations with constraints that didn't contain binary variable, still for the final solution this library

was unsuitable. For these reasons, the description will be provided only for three libraries – PuLP, CVXPY, and Pyomo.

Furthermore, before applying any libraries solvers the data should be pre-processed. For the further libraries described, the data were prepared in a similar format. Initially the dataset was loaded to dataframes, and variable names were stored into dictionary. After several data adjustments, such as adding columns and changing data types, additional dataframes were created, for supply and demand, in order to conveniently access information when building constraints. In order to speed up the further process of calculation and solvers applications, all dataframes were transformed into dictionaries. An example of such one, which was used, was a dictionary, where keys are numbers of products, and values contain the names of products, and to which product group they relate. In addition, the dictionary contained another dictionary called "product transportations" which represented on information on the transportations of the concrete product – from which oil base to which petrol station, the volume and cost of transportation. As a result, the dictionary replaced all information on transportations which was stored previously in dataframe. Again, it should be noted that if in the current month for a trio of <oil base-petrol station-brand> the information on tariffs was given, but no matching <oil base-product> or <petrol station-brand> were found in supply and demand (or if found values were equal to zero), such connections were deleted.

2.7.1. PuLP implementation

As was discussed in the literature review chapter, PuLP is a linear-programming modeler in Python, which allows access to the following solvers for MIP problems: CBC, MIPCL, SCIP (Vigerske & Gleixner, 2018). To solve the transportation problem using PuLP, the model should be initialized first, and the type of problem (minimization) should be specified. Then, the variables should be listed. In case of the current problem, the variables are identified as $P_n O_{m\_} brand_k$, which is the volume of brand k that can be delivered from an oil base m to the petrol station n. The objective function is defined by means of affine expressions, containing variables for volumes and associated costs. Both volumes and costs are represented in a simple form of variable.

Next, the constraints are to be specified. In the case of PuLP, the constraints are input in a very user-friendly form, literally by writing an equation in a classical form. For instance, for the purpose of adding a constraint on demand, affine expressions are used, specifying that each of volume variable for each petrol station should be equal to some constant number. In a similar way all other constraints are listed, and the model is formulated. In PuLP the model is represented in a very readable way: the objective function and constraints are shown in s standard form of equations, then for each variable

its type is shown (for example, $P_0O_0\_brand_0$ is continuous). Then the solver is chosen (for PuLP the default solver is CBC), and the solution is built, listing all variables and assigned volumes to them (how much petrol of each product is delivered from each oil base to each petrol station).

The required time for finding an optimal solution for the current problem for three regions turned out to be 9.86 seconds with CBC solver (139.6 million rubles). Overall, PuLP is a convenient and flexible library for the problem considered because it allows a variety of instruments depending on the problem type. In addition, there is no need to create complex scripts for data input, because PuLP works with a very intuitive form of data, compared with matrixed.

2.7.2. CVXPY implementation

For CVXPY the overall logic is similar to PuLP's one. It also takes as an input a set of equations for objective function and constraints. As well as PuLP, CVXPY uses laconic form of affine expressions for specifying equations, which also simplifies the syntax and improves the readability of code. However, the representation of the minimization function and constraints is less readable than in the case of PuLP. As was mentioned before, PuLP lists everything in the form of equations with concrete variables, thus it is effortless to check the correctness of the model. As for CVXPY, it lists only the fact of the expression (equality or inequality) and the type of its parts (nonnegativity, constant variable), without names of variables and values. Therefore, it is hard to be sure that all equations are written in the correct form and that all constraints are taken into account. In addition, CVXPY is different from PuLP in a way that it does not allow user to access the variable by its name, it uses index, which is an integer number. That in turn complicates user interaction with a model and makes it difficult to implement dictionaries with key data structures. The time needed to produce the result is 11.32 seconds with default CBC solver (143.3 million rubles).

2.7.3. Pyomo implementation

As for the previous two libraries CVXPY and PuLP, the implementation logic in Pyomo is about the same, also without need for matrix form of data. First, the model is initialized, and a variable is declared, which will include a set of transportation volumes for each type of product transported from different oil bases to petrol stations. The values of this variable can then be accessed by index, which is convenient to implement, but creates difficulties in interpreting finite numbers. These variables are set as positive real numbers.

Next, the objective function in Pyomo is declared almost as laconically as in CVXPY, which also means simple syntax and code readability despite the of lack of affine expressions. Then it is separately clarified that objective expression is to be minimized. After that, a list of constraints is

created where equalities and inequalities are added in turn for all four constraint types mentioned above. There is an opportunity to view the contents of the constraints, but it looks incomprehensible, and the desired variables are designated as referring to a variable by index, so it is difficult to understand which product, oil base and petrol station the presented values belong to.

The same problem is encountered when obtaining a solution to the optimization problem. The answer in solution is the total cost of transportation of petroleum products, as well as a list of total volumes to be transported, and dummy variables (to fulfill «one to one» limitation, where 0 is when products are transported only from the oil base number 0, and 1 is when from the oil base number 1). Since these volumes and dummy variables are presented simply as lists, there is a need for further transformation of the response to be able to correlate with specific products represented as $P_n O_m\_brand_k$. As for the addressing variables, Pyomo has similar problem as CVXPY, requesting to use indexes instead of variables names.

It is worth noting that the default solver in this Pyomo library for the MILP problem is also CBC. As for the time to find the optimal solution, the solution is slightly faster than the previous two libraries and is approximately 6.17 seconds with CBC solver (182.4 million rubles). Also, a free Bonmin solver can be potentially used in this library to solve the MILP problem.

In the next chapter, the information about the convenience of technical implementation of libraries is summarized based on the set of certain criteria, which have already been partially discussed here.

2.7.4. User interaction with program

Here, the main principles of user interaction with program in terms of input data, and output results, are described.

First of all, it should be noted that the program has a functional form, meaning that all interaction is organized in a form of functions with input parameters and return value. There is also a configuration file in which the user can specify constants. The constants include the name of the file with the source data, the month, and the set of regions for which it is necessary to calculate the distribution. Overall, the normal case is when the code is loaded on all regions because there is no strict connection between a pair of <oil base-petrol station> and single region, however, it may be the case when company wants to regulate this parameter and choose concrete regions. Also, it is a good parameter for quick testing of solvers. Configuration file also contains the loading threshold of the oil base, types of upper and lower bounds of supply's restriction (by group of products or by product), and origins to control which means those oil bases for which the lower bound is set. It is done in order

to provide flexibility of the program if some adjustments and changes are done in future. The description and example of configuration file content is represented below in the Table 7.

Table 7. Configuration file description. Source: (Provided by authors, 2023)

Parameter	Description	Possible choices	Example
INPUT_FILE_NAME	Name of the input file	.csv/.xlsx/.xlsb file	'opt_data_gen.xlsb'
MONTH	Month to be chosen for model building	Integer from 1 to 12	1
REGION	Regions to be chosen for model building	List of numbers of regions	[1, 2, 4]
PERCENT_LOADING	Percentage of minimum loading of oil base	Decimal between 0 and 1	0.2
M	Big number for linear transformation	Any real positive number, which is big enough	100 000 000
ORIGINS_TO_CONTROL	Origins for which percent loading is set	List of numbers of origins	[10, 13]
UPPER_BOUND	Type of upper bound (either for product or for group of products)	'group' or 'product'	'group'
LOWER_BOUND	Type of lower bound (for product, all products, or for group of products)	'group', 'product', or 'all products'	'product'

After inserting necessary constants in configuration file, the program can be launched. It translates the source data into the desired form for further work of solvers, by changing data type, creating dictionaries, and splitting data into demand and supply sides.

Then, the program applies one of the solvers in order to calculate the optimal distribution of volumes between petrol stations and oil bases. Finally, it prints the values of volume for each trio point-origin-product, the resulting cost with such distribution, and time spent of the calculation.

2.7.5. Usage of solvers

In more detail about the solvers, there are various solvers inside each library that may differ from each other in several aspects (Karlof, 2005).

- First of all, there are various methods inside solvers, that is, *algorithms* of work that solve the optimization problem. As described above in the literature review, the main algorithms for solving MILP problems are the Branch and Cut method, the Branch and Bound method, and the Interior-point method.
- The *speed* of the solvers also differs depending on the algorithm lying inside it. When there is a lot of data, then the speed of the algorithm is especially important. It is also important to keep in mind that as a rule commercial solvers work faster (Karlof, 2005).

- The *accuracy* of optimization results is also slightly different. Some solvers are better able to approach the real optimum, that is, more accurate. It is worth mentioning that this aspect and the previous one about speed are the most important for comparing the selected solvers.
- As for *support*, some solvers have better quality, are updated and have extensive documentation. Others, on the contrary, are already outdated, such as the free MINTO (Nemhauser et al., 1994) solver in the Pyomo library. It used to be quite popular to solve MILP problems, but now its functionality has been applied in other solvers, for instance, in CPLEX which is commercial solver for MILP. Because of this MINTO is no longer supported, so it has not been considered in this paper. In the table below there are only solvers that are supported now.
- As for *licensing*, access to solvers is different, some of them are available for payment, some are free (Meindl & Templ, 2012), which was reflected in Figure 10. For this project, only those with free access were used.
- The next significant aspect is *compatibility*. It should be borne in mind that some solvers are compatible only with certain operating systems, programming languages or platforms.
- And the last aspect is the *user interface*. Some solvers have a user-friendly interface, that is, the Graphical User Interface (GUI) is used. Others may require more technical knowledge to access them via the Application Programming Interface (API) or via the Command-line interface (CLI).

  These aspects should be considered while choosing a solver.

2.7.6. Experiments

For the sake the saving computational time, it was decided to use a small test data set, which contained only three regions; we use the results the obtain to design the final solution.

Experiment with supply restrictions

It was decided to test different variants of supply with upper and lower bounds, setting them for group of products (petrol or diesel), for each product separately or for sum of all products. In order to test and compare those options, the PuLP library was selected, and only first month and several regions were chosen (1, 2, and 4 region). There were in total six combinations of lower and upper bounds: lower and upper by group, lower by group and upper by product, lower by product and upper by group, lower by all products and upper by group, lower and upper by products, lower by all products and upper by product.

As a result, with the percentage of loading of 20%, the following results were obtained. The highest costs (144.9 million rubles) were obtained with following options: upper and lower by group, upper and lower by product. Then 144 million rubles was shown when upper bound was by product and lower by group of products or by all products. When the supply constraints were upper by group and lower by product, the total cost was 139.7 million rubles. Finally, the lowest value (139.6 million rubles) was received by applying lower bound by all products and upper by group. Thus, for further testing it was decided to use this particular combination of supply bounds. Probably high costs values were received when constraints were too strict (such ad by concrete products), and the lowest cost were when products were grouped somehow. In this case the origins for which there was percentage loading are origins with ratio of supply to demand in Table 6 was below 0.5. We do not consider all products as upper supply bound due to industry specifics. Percent loading starts to influence the value of the objective function only with sufficient value, such as 0.5 or above. This information is summarized in Table 8 below.

Table 8. Comparison of results with different supply bounds. Source: (Provided by authors)

Supply upper bound	Supply lower bound	Values of functions, million rubles
Group	Product	139.7
Group	Group	144.9
Product	Product	144.9
Product	Group	144
Product	All products	144
Group	All products	139.6

Experiment with solvers and libraries

Summary information about tested libraries and successful free solvers inside them is presented in Table 9 below.

Table 9. Comparison of results by solvers on 3 regions. Source: (Provided by authors, 2023)

Library	Solvers	Time running for solution (seconds)	Result (rubles)	Methods inside	Time running for constraints' creation (seconds)
PuLP	CBC	9.86	139 625 221	Branch and Cut	10.77
	SCIP	10.79	137 652 045	Branch and Bound	11.72
CVXPY	CBC	11.32	143 300 000	Branch and Cut	21.12
Pyomo	CBC	6.17	182 427 606	Branch and Cut	13.36

In Table 9 there is a comparison of successful solvers inside libraries by the amount of time during which the optimization problem was solved, then by the result obtained (only for 1 month, in order to quickly test experiments) and methods applied inside the solver. As for other important conditions, supply upper bound by group was chosen and no origins to control were specified; therefore, there is no lower bound for supply.

One may notice that in Figure 10 more chosen were present for MILP problem; below in Table 10 more detailed commentary on each of these solvers is provided.

Table 10. All solvers considered. Source: (Provided by authors, 2023)

Library	Solver	Methods	Explanation
PuLP	CBC	Branch and Cut	Successful installation and result
	SCIP	Branch and Bound	Successful installation and result, but sensible to problem complexity
	MIPCL	Branch and Bound	Not installed: not available on the internet
CVXPY	COPT	Branch and Bound	Successful installation and result, but embedded restriction on the number of constraints
	CBC	Branch and Cut	Successful installation and result
	ECOS_BB	Interior-point	Successful installation and result, but highly unreliable and uneven performance
	GLPK_MI	Branch and Bound	Successful installation, but only works on a very small task
	SCIP	Branch and Bound	Available through a package that requires higher version of operation system
Pyomo	CBC	Branch and Cut	Successful installation and result, but bad accuracy of result
	BONMIN	Branch and Bound	Successful installation, but only works on a very small task

So, with some of the solvers two types of failure were met. Firstly, some solvers could not be successfully installed: MIPCL (because it was not available for installation on the Internet) and SCIP for CVXPY (as in CVXPY it requires OS with more resources in order to install this package). Both these solvers are based on Branch and Bound method.

The second type of failure are the solvers that were successfully installed but were able to solve only small similar tasks of the same type of MILP problems. However, they cannot cope with the current task that is a more complex task since they seem to be not powerful enough. There are such solvers as COPT, ECOS_BB, GLPK_MI (from CVXPY library) and BONMIN (from Pyomo library). All the listed solvers except ECOS_BB are based on the Branch and Bound method. It is

worth noting that ECOS_BB solver based on Interior-point method works unstable and may show different results for some reason, sometimes failing to solve the problem by choosing infinity as the solution. COPT has data volume restrictions and also does not solve the current more complex problem.

2.7.7. Final results

There was an attempt to apply the SCIP solver of the PuLP library to all regions, but this solver shows unstable results with a large number of restrictions, sometimes failing to deliver the results. As for Pyomo CBC, it was shown in Table 9, that it has the highest costs, so it was probably not the most optimal solution. The general observation that was made throughout all the experiments with the model is that they can show inconsistent results; in some cases, the solver in the same library shows different results at different runs. Probably the reason behind this behavior is that there might be a stochastic component in how the whole optimization process runs, for instance, the initial distribution of volumes; the results may be also influenced by the environment (RAM, CPU) in which the program is launched. Therefore, the user is encouraged to try different versions of the model developed in this paper if he or she seeks to find the most optimal solution in his or her case.

It also should be mentioned that no lower bound for supply is specified in the final solution; representatives from Gazprom Neft Company have explained during the consultation that the lower bound of supply is specified only for a small set of oil bases, which is up to a user.

As a final result the PuLP CBC solver was chosen, as it performed the lowest time with accurate results on the test piece of data. Thus, it was applied on the final dataset with the following restrictions. Firstly, there were all regions chosen and only 1 month. Secondly, as was mentioned in the experiments with solvers part, the upper bound was chosen to be for group of products, and the lower bound was for all products but without origins to control. The model has shown the following: 274 seconds, or 4.5 minutes, on restrictions insertion; 1282 seconds, or 21 minutes, on the model delivering a solution; and the result obtained equaled to 125.9 million rubles, which is lower than both planned (264.4 million rubles) and actual (304.1 million rubles) baselines. The total number of constraints was 20 742, the total number of variables: 67 894, including volume variables and 14 092 dummy variables.

Another solver that was successfully applied on the whole dataset was CBC solver from CVXPY library; it has shown worse performance in terms of time; it took 315 seconds, or 5.25 minutes, to make the restrictions, 2306 seconds, or 38 minutes, to deliver a solution. The optimal value, however, was significantly lower in terms of costs: 84.7 million rubles. Though this low value

was unexpected, the check-ups has shown the restrictions were indeed upheld correctly. The total number of constraints and the total number of variables were the same as in PuLP CBC described above. It should be noted that the value of the final solution created on the whole dataset can be below the solution on the test dataset (Table 9) because system is optimized as a whole.

2.8. Summary of chapter 2

At the beginning of this chapter, the description of secondary data provided by Gazprom Neft company was given. The dataset included several key parameters like oil bases, petrol stations, products, brands (made from products); some cost-related parameters like tariffs, transportation fleet, and distance; demand and supply parameters: volume of demands and volume of supply, date (for which information is provided), region, and baseline. More information is available in Table 2.

In addition, the mathematical model of the current transportation problem was formulated. The objective function is supposed to minimize operational expanses on logistics; four restrictions are: equality constraint on demand, upper and lower inequality constraints on supply, and «one-to-one» constriction which says that all brands are to be delivered from only one oil base. The latter constraint is realized with the usage of a dummy variable. For more details on mathematical model refer to formulas from 15 to 22 which specify: objective function, constraints, constraint on variables, and description. Apart from that, possible variations of the mathematical model were described.

Next, the type of optimization problem was identified as a Mixed Integer Linear Problem, or MILP; in short, the reasoning behind this conclusion is the following: all the equations are of linear nature, and there are not only continuous, but also discrete variables; therefore, the optimization problem is non-convex, datatype is mixed, equations are linear, a convex relaxation is used. For more details see information present in Figure 9.

Furthermore, based on the type of optimization problem, suitable technical tools (libraries, packages, and solvers) were chosen, see Figure 10. Different versions of the code for each library/package were developed; up to the present moment, these working technical implementations include versions in PuLP, CVXPY, and Pyomo. In this chapter only a brief comparison of the packages/libraries was provided that is to be expanded in the following chapter.

Next, some experiments with variations of mathematical model were performed; the respective results can be seen in Table 8. Overall, six combinations for supply constraints were compared; expectedly, the combination of upper bound by group and lower bound by all products provided the best result in terms of overall costs. Then, comparison of the results yielded with the usage of different solvers was present in Table 9; main criteria included time running and objective function's value

which is overall costs. Four cases were considered as successful: CBC and SCIP for PuLP, CBC from CVXPY, and CBC from Pyomo. As for the other solvers, detailed commentary on each was provided in Table 10.

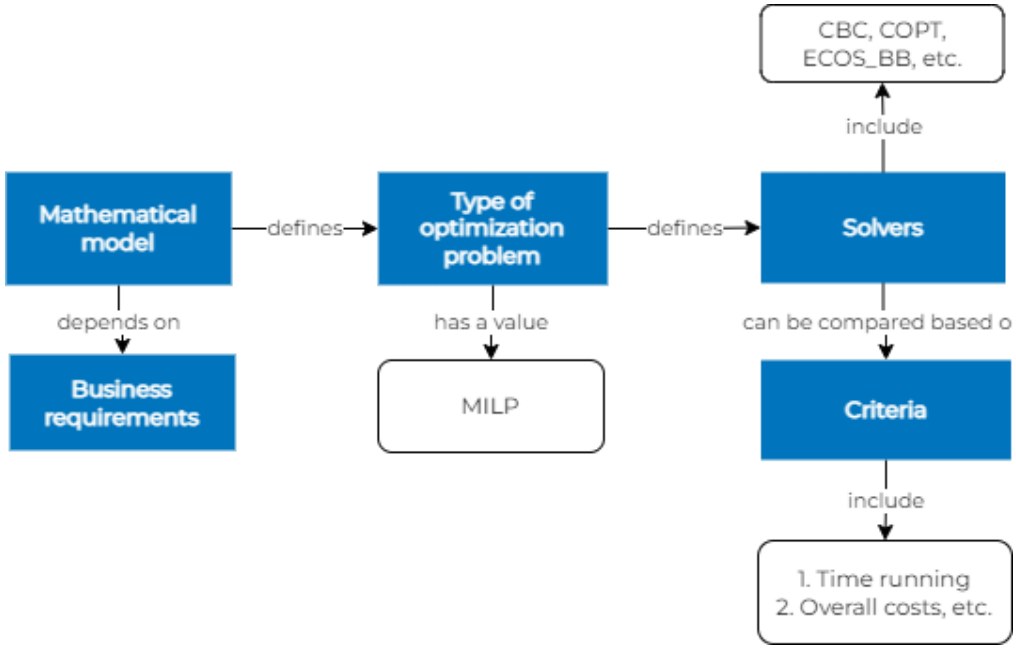


Figure 11. Concept map of Chapter 2. Source: (Provided by authors, 2023)

Finally, results obtained on the whole dataset were present; CBC solver from PuLP was chosen based on the experiments on test dataset. The result provided by the model was less than the baseline specified by the company by 138.5 million rubles for planned budget and 178.2 million rubles for actual values. Also, CBC from CVXPY was implemented on the whole dataset with relatively the same performance in terms of speed and lower costs. However, one should be cautious with such positive results because previous version of the baseline proved to be inaccurate, and probably the same can be stated for the latest version we compare our solution with. All in all, the solution provided by the model can be seen as a starting point for further managerial decisions.

Main logical elements of this chapter and connections between them are present above in Figure 11. To sum up, mathematical model is developed based on the business requirements; it defines a type of optimization problem, MILP in this case. Type of optimization problem, in term, defines the choice of solvers; these include such solvers like CBC, COPT, ECOS_BB, and others, solvers can be compared based on criteria which may include such parameter like time running, objective function's value, etc.

CHAPTER 3. ANALYSIS AND EVALUATION OF RESULTS

3.1. Comparison of results based on developed criteria

In the first theoretical chapter, the types of optimization problems were studied, taxonomies of problems and ways to solve them were formed, and knowledge about the possible technical implementation of possible solutions using well-known libraries and solvers in Python was summarized. Then, in the second methodological part, a specific transportation problem for Gazprom Neft was considered based on their data, a mathematical model was built and technical implementation of suitable tools for solving the MILP problem was carried out.

Next, in this chapter the technical results obtained should be analyzed. Criteria should be developed based on which decision is to be made on which specific library and solver should be applied in Gazprom Neft for the transportation problem solution.

3.1.1. Comparison of libraries based on developed criteria

A certain set of criteria was derived by which it is possible to compare the applied libraries to the solution of the current problem. In this section, a description of each criterion is presented in turn with a description of their weight (from 0 to 1) among all criteria. Each criterion is measured on a scale from 1 to 5 for all libraries, how well these criteria are fulfilled for them. In the scale 1 is least developed in the criterion, 5 is the most developed in the criterion. Then the weights of the criteria were multiplied by the score for each library, so that each received its own final score, on the basis of which a decision can be made. The chosen criteria for libraries' comparison are described below. Criteria for libraries include flexibility, interactivity, and quality of documentation.

Flexibility

It denotes the versatility of the library for solving problems of different types. All the libraries discussed here are suitable for solving the current MILP transport problem with four business constraints described in the methodological part in formulas 8-14. But the conditions of constraints may change, the task may become, instead of MILP, an optimization problem of some other type from the presented taxonomy of optimization problems in Figure 3. Flexibility means whether this library can still solve the problem if some conditions change. This criterion also includes the availability of solvers in the library, whether there are many of them that can be successfully installed and applied to the current task.

The weight of flexibility among the criteria is 0.5. It is the most important criterion since it directly influences the business decision to use this or that library and to scale it. The calculation for this criterion follows in formula 26:

$$Flexibility_i = \frac{FS_i}{max(FS_1, .., FS_i, \dots)} \times \frac{P_i}{\max(P_1, .., P_i, \dots)} \times \frac{(1 + S_i)}{(1 + D_i)}, for \forall i \quad (26)$$

Here i is the index of the library, FS is the number of free solvers inside the library, P is the number of problems that can be solved inside the library with free solvers, S is the number of successful installation of solvers that work on the task for a library, D is the total number solvers tried for installation for the library. After the calculations are carried out, the maximum score (5) for the criterion is allocated to the library with the highest flexibility, all others receive relative scores.

Interactivity

Interactivity relates to the convenience of working with the library. This criterion means the easiness of syntax perception which means readability of the code, the easiness of input data format, and the interpretability of intermediate results, which includes final results as well.

Firstly, syntax which is readable and not overcomplicated is preferrable for users who work with library. This is a significant criterion that affects the speed of development and code support. Secondly, when there is a lot of data, restrictions, etc., data format is important. It is about how easy it is to input data to the problem, to change them, etc. For example, if the library requires input data in the form of matrixes, then it is inconvenient to compile matrix data and if something needs to be changed in them, then it also requires a lot of time and effort. Then, as for intermediate results perception, it is important in what form the conditions (optimization function, constraints, information about variables) are stored in the problem function, whether it is possible at any time to view the saved conditions in a readable format. Finally, the solution that the library gives is important. It is desirable to immediately see which pairs of oil bases and petrol stations have been assigned the received volume values, without further refinement. The weight of interactivity in the criteria is 0.3 because it should be easy and clear to interact and to have low probability of error. It is important to note that all the sub-criteria here is evaluated subjectively based on the authors' experience with given libraries. The calculation for this criterion follows in formula 27:

$$Interactivity_i = Syntax\ Perception_i \times Data\ Format_i \times Intermediate\ Results_i, \quad for \forall i \quad (27)$$

Here i is the index of the library, $Syntax\ Perception$ means the easiness of syntax perception which means readability of the code and is evaluated from 1 to 5, $Data\ Format$ means the easiness of input data format and is evaluated from 1 to 5, and $Intermediate\ Results$ means the interpretability of intermediate results, which includes final results as well, and is also evaluated from 1 to 5. After the calculations are carried out, the maximum score (5) for the criterion is allocated to the most interactive library, all others receive relative scores.

Quality of documentation

The availability of detailed quality documentation is significant for someone who works with the library. Quality documentation allows users to explore the functions and capabilities of the library efficiently and quickly. If the conditions of the task change, some more library features may be required, the availability of additional information simplifies the work. This criterion is essential but slightly less important than the previous ones, so its weight among the criteria is 0.2, because an expert can be found and consulted on some ambiguous questions on documentation. In Again, the evaluation is more or less subjective, especially in what concerns the intelligibility of the documentation, The calculation for this criterion follows in formula 28:

$$Quality\ of\ documentation_i = Intelligibility \times Completeness\ , for \forall\ i \tag{28}$$

Here i is the index of the library, $Intelligibility$ means easiness of documentation perception: if the documentation is communicated in a clear and understandable manner; intelligibility is evaluated from 1 to 5. $Completeness$ of the documentation includes the availability of relevant objects and methods description and is evaluated from 1 to 5, as well as the availability of examples and accuracy of information in the sense that it is up to date. After the calculations are carried out, the maximum score (5) for the criterion is allocated to the library with the most quality documentation, all others receive relative scores.

Overall assessment for library

The Table 11 shows the summary information of formula application to each library. In fact, this table contains all the relevant information on the comparison of the libraries in a short form; consequent Tables 12 and 13 are present for convenience and justification of evaluation purposes.

Table 11. Calculation of criterion. Source: (Provided by authors, 2023)

	Formula	PuLP	CVXPY	Pyomo	SciPy
Overall total on library		**3.32**	**4.04**	**3.72**	**1.44**
Total on flexibility		2.05	5.00	3.64	1.02
Flexibility	$\frac{FS_i}{max(FS_1,..,\ FS_i,\dots)} \times \frac{P_i}{\max(P_1,..,\ P_i,\dots)} \times \frac{(1+S_i)}{(1+D_i)}\ , for\ \forall\ i$	0.27	0.67	0.48	0.14
FS	The number of free solvers inside the library	4	11	8	4
P	The number of problems that can be solved inside the library	4	4	4	3
S	The number of successful installation of solvers that work on the task for a library	2	3	1	0
D	The total number solvers tried for installation for the library	3	5	2	1
Total on interactivity		5.00	3.00	3.00	0.96
Interactivity	$Syntax\ Perception_i \times\ Data\ Format_i \times Intermediate\ Results_i, for\ \forall\ i$	1	0.6	0.6	0.19
Syntax perception	The easiness of syntax perception which means readability of the code (1-5)	5	5	5	4
Data Format	The easiness of input data format (1-5)	5	5	5	3
Intermediate Results	The interpretability of intermediate results (1-5)	5	3	3	2
Total on quality of documentation		4.00	3.20	5.00	3.20
Quality of documentation	$Intelligibility \times Completeness\ , for\ \forall\ i$	0.8	0.64	1	0.64
Intelligibility	The easiness of documentation perception	4	4	5	4
Completeness	The availability of relevant objects and methods description	5	4	5	4

Table 12 below contains a more detailed description behind the reasoning based on which libraries' scores were previously allocated in Table 11. Here more attention should be paid to criteria of interactivity and quality of documentation since these are more subjective.

Table 12. Justification of score for libraries. Source: (Provided by authors, 2023)

Library	Justification of score for library	Score
Flexibility		
PuLP	It is suitable for solving LP, NLP, MINLP, MILP problems, there are various solvers, both free and commercial; there are 4 free solvers in total. If the conditions of the problem change slightly, then there is no need to switch to another library. Out of all the solvers suitable for the task, only 2 out of 3 solvers was successfully downloaded and worked for the current task.	2.05
CVXPY	This library can also be applied to solve LP, NLP, MINLP, MILP problems; in CVXPY there is a maximum number of free solvers available among other libraries — 11 in total. Out of 5 solvers, 3 worked on the provided task.	5.00
Pyomo	Suitable for solving LP, MILP, NLP, and MINLP problems. There are 8 free solvers available for solving those problems. Out of 2 solvers, which are suitable for the considered case, 1 was successfully installed and applied.	3.64
SciPy	It is suitable for solving LP, NLP, MILP problems, but not MINLP, which is a disadvantage. Also, the variety of solvers is not as large as for CVXPY and Pyomo libraries. There are 4 free solvers in total, 1 of them is available for MILP, and there are no commercial solvers available.	1.02
Interactivity		
PuLP	The data format in which restrictions are introduced into the model has a very user-friendly format, data can be fed into the model from a dataframe. In general, syntax is not hard; the model and constraints are shown in a readable format, it can be always checked what exactly is written there, what are the constraints, what type of variables, etc. The optimization result that the model gives is easy to understand and interpret, because it is written which volumes relate to which specific products, from which oil base they were delivered and to which petrol station.	5.00

Library	Justification of score for library	Score
CVXPY	The form in which restrictions are introduced into the model also has a user-friendly format, data as well can be fed into the model from a dataframe, not matrix, and syntax is not overcomplicated. But compared to PuLP, it does not work here to check how the model and constraints are written inside. And also, the disadvantage is that it is not clear what volumes of products were obtained for which pairs of oil depots and petrol stations, this requires a certain decoding.	3.00
Pyomo	For syntax perception and data format - the same as for CVXPY. But the difference from CVXPY is that the result of the solution is a little more understandable, although additional decoding is also required; and there is possibility to check how the model, variables and constraints are written inside.	3.00
SciPy	Working with this library is quite complicated in terms of data structures, this process is difficult to automate and requires some serious data preparation. Data matrixes are required for input, it becomes difficult to compile them for the case when there are a lot of products, oil bases, petrol stations. Even for the amount of data that is used in this project, the task is time-consuming and there is a high probability of making an error when compiling matrixes (especially when scaling a project). In addition, the conclusion that is given as a result of optimization requires additional decoding, since the array of data with volumes goes without reference to which product, oil base and petrol station they belong to.	0.96
Quality of documentation		
PuLP	There is extensive documentation with examples for the library, which is regularly updated as well as the library itself (Pulp Documentation, 2023). However, some topics are not described in detail and have just reference to some other parts of documentation.	4.00
CVXPY	Also, there is extensive documentation for this library with many examples for the MILP problem (CVXPY, n.d.). However, it is not fully updated, for example, information not about all solvers is added to it.	3.20

Library	Justification of score for library	Score
Pyomo	It is the most extensive and understandable documentation for this library, which is regularly updated as well as the library itself, many examples for the MILP problem (Pyomo Documentation, 2023).	5.00
SciPy	In general, the documentation is complete and contains some usage examples describing function parameters, formulas, notations. It is regularly updated, but it is not as detailed as documentation for other libraries, for example, there are not so many descriptions and examples (SciPy, n.d.).	3.20

As was stated previously, weights for the criteria were assigned in accordance with the importance of the factor for working and obtaining a ready-made solution with the library. Weights were the following: $Flexibility$ — 0.5, $Interactivity$ — 0.3, $Quality\ of\ documentation$ — 0.2. The sum of all the weights in total gives 1. Formula 29 describes overall assessment of the library:

$$Total_i = 0{,}5 \times Flexibility_i + 0{,}3 \times Interactivity_i + 0{,}2 \times Documentation_i, for\ \forall\ i \quad (29)$$

Here i is the index of the library. Table 13 below summarizes the results of comparing libraries by criteria.

Table 13. Summarized comparison of libraries. Source: (Provided by authors, 2023)

Library	Flexibility (0.5)	Interactivity (0.3)	Quality of documentation (0.2)	Total
PuLP	2.05	5.00	4.00	3.32
CVXPY	5.00	3.00	3.20	4.04
Pyomo	3.64	3.00	5.00	3.72
SciPy	1.02	0.96	3.20	1.44

In Table 13, the weighted sum for each library is calculated in the Total column according to the criteria presented above. According to this table, the most suitable libraries are PuLP, Pyomo, CVXPY (their totals are equal to 3.32, 4.04 and 3.72 respectively). SciPy library here is the least suitable, because of this it has not been fully applied in this project (total is 1.44).

3.1.2. Comparison of solvers based on developed criteria

Then, for chosen libraries there is a comparison based on solvers. For the applied solvers, their own set of criteria was also developed, also on a scale from 1 to 5, and each criterion again has its own weight from 0 to 1. The chosen criteria for solvers' comparison are described below. Criteria include speed and accuracy of result.

Speed

It is crucial for the company to get results quickly. When there is not a lot of data provided, there is not much difference in speed. But with an increase in the data set, if there is information on oil bases and petrol stations in a large number of regions, then the difference in the speed of the solvers becomes noticeable. This criterion is slightly higher priority than the second one and has a weight of 0.6.

Accuracy of result

This criterion is about how accurately the minimum of the objective cost function was found. It is significant that the obtained optimal cost value is approximately in the area of the baseline solution. It should be considered that the baseline may be imprecise, since it was calculated manually by specialists, also discrepancies may arise due to a slight change in tariffs. The criterion has a weight of 0.4, because the result may be approximately correct, but not absolutely accurate. Table 14 summarizes the results of comparing solvers by criteria. Only those solvers from Table 9 that worked successfully were assessed here. Solvers without ratings are those solvers that were not used in the work, the reasons for this were explained in more detail in the second chapter in Table 10.

The speeds of implemented solvers were compared with each other in accordance with the results of Table 9 that was built for three regions only. The fastest solver has a score of 5, the rest of the scores are set relative to it. Further, accuracies of the results were compared with the baseline solution for the first month. Those solutions with higher accuracy (they have similar results to each other) have a higher score on a scale from 1 to 5.

Table 14. Summarized comparison of solvers. Source: (Provided by authors, 2023)

Library	Solvers	Speed (0.6)	Accuracy of result (0.4)	Total
PuLP	CBC	5.0	5.0	5.0
	SCIP	4.0	5.0	4.4
CVXPY	CBC	4.0	4.0	4.0
Pyomo	CBC	5.0	3.0	4.2

In Table 14, the weighted sum for each solver is calculated in the Total column according to criteria of speed and accuracy of result presented above. According to this table, the most suitable solver is CBC from PuLP library (total is 5.0). Other solvers are worse in terms of speed and accuracy but still have good enough results in Total (4.4, 4.0 and 4.2 for SCIP from PuLP, CBC from CVXPY and CBC from Pyomo respectively).

3.1.3. Choice of the most suitable library and solver based on developed criteria

In this section the results of comparison obtained previously are combined to choose the best solution(s). In this paper more weight is assigned to the solver (0.6) because the speed of the solution and its accuracy were highlighted as the crucial factors for the choice of the solution by the company. The weight allocated to the library is, respectively, 0.4. In Table 15 below choice of the final solution is presented.

Table 15. Choice of final solution. Source: (Provided by authors, 2023)

Library	Total score for library (0.4)	Best solver	Total score for solver (0.6)	Total
PuLP	3.32	CBC	5.0	4.328
		SCIP	4.4	3.968
CVXPY	4.04	CBC	4.0	4.016
Pyomo	3.72	CBC	4.2	4.008

Based on a comparison of the used libraries and solvers using the developed criteria, it can be concluded that the most suitable solution is CBC solver from PuLP library. This CBC solver is easy to implement and works good in terms of accuracy and speed. As for the libraries, Pyomo is beneficial in the aspect of the quality of documentation, CVXPY — in terms of flexibility and PuLP — in terms of interactivity.

3.2. Approach to visualization of results

The proposed model gives three main advantages over the previous model that existed in the company earlier: optimization of operational expenses, time saving and saving of human efforts. Overall, the business problem stated in the beginning of the thesis work concerned optimizing operational expenses on secondary logistics. The described model is aimed at minimizing these expenses and turned to be less than the provided by company baseline. Moreover, as the model works automatically, it saves human efforts, because now a decision-maker should only provide input data and set necessary parameters instead of calculating all costs manually. The model is flexible because it allows user to

specify some parameters without any additional time spend. For example, the user can choose the oil bases on which to set a lower bound depending on the current business requirements.

In addition, in the beginning of the project there was a requirement about time, stating that the model should calculate the optimal cost in around 25 minutes. The proposed model brings value of time savings because all of the solvers showed fast results on the test piece of data for only three regions and one month. With increasing of the data amount, the time had somehow increased, but it is still around specified 25 minutes. More information on time running for three regions was specified in Table 9 and in final results' paragraph.

In this section the approach is proposed that can be applied to the model's results. Here the visuals are built for the test sample of three regions. All in all, these plots are meant to underline the following:

- **Business and technical value:** The graph of the values of the objective function showing monetary savings (line graph) and the graph of the dependence of the calculation time on the volume of the source data;
- **Solution:** The optimal distribution of volumes between oil bases and petrol stations in a form of network;
- **Robustness of solution:** Sensitivity analysis with specification of values of the objective function depending on the loading of the oil base;

Considering all the parts described above, figures were created using Python. As an illustrative example the PuLP library was used, in particular one solver was considered, which is PULP_CBC_CMD. The user can use the code proposed to create his or her own visualization of the results.

3.2.1. Business and technical value

Here and below all the visual examples are given for a test piece of data, which contained only three regions: 1, 2, and 4. This was done for the sake of saving computational time. Here, it can be seen in Figure 12 that costs from the solution provided by the model are about 15% less than costs of the baseline solution; and working time of solver is less than 7 seconds. Obviously, it is not entirely correct to compare solution on the partial dataset to the respective baseline on three regions, but here it is done for the sake of the example.

Figure 12. Example of indicators of costs and working time for the first month. Source: (Provided by authors, 2023)

Figure 13 below shows the relative difference earned in percent with the usage of the model for two first months; here the difference between baseline and solution costs goes in nominator, and baseline costs go in denominator.

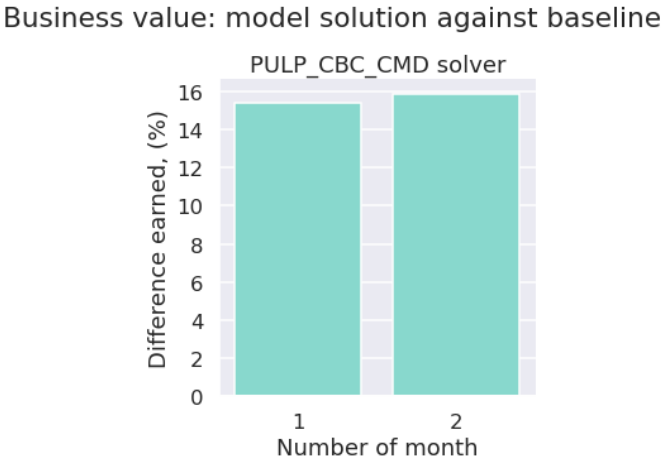


Figure 13. Model solution against current baseline for the first month. Source: (Provided by authors, 2023)

3.2.2. Solution

Figure 14 below shows the main incoming information about the model for a concrete month: the number of oil bases, petrol stations, brands and products, as well as the overall number of pairs (oil bases, petrol stations, and brands) and pairs connected as the result of the model.

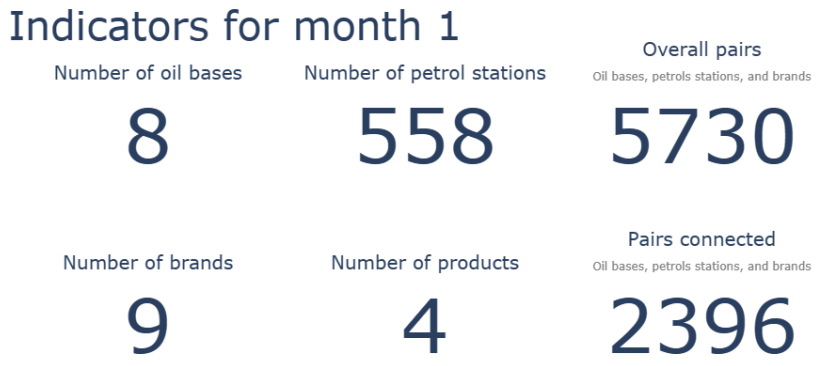


Figure 14. Example of indicators for the first month. Source: (Provided by authors, 2023)

The Figure 15 below illustrates the solution for the CBC solver, representing the pairs of oil bases and petrol stations for the first month. It can be seen that some oil bases like oil base 10 and oil base 13 provide more connections with petrol stations than other oil bases. There are only four oil bases on the graph because they are provided just as an example.

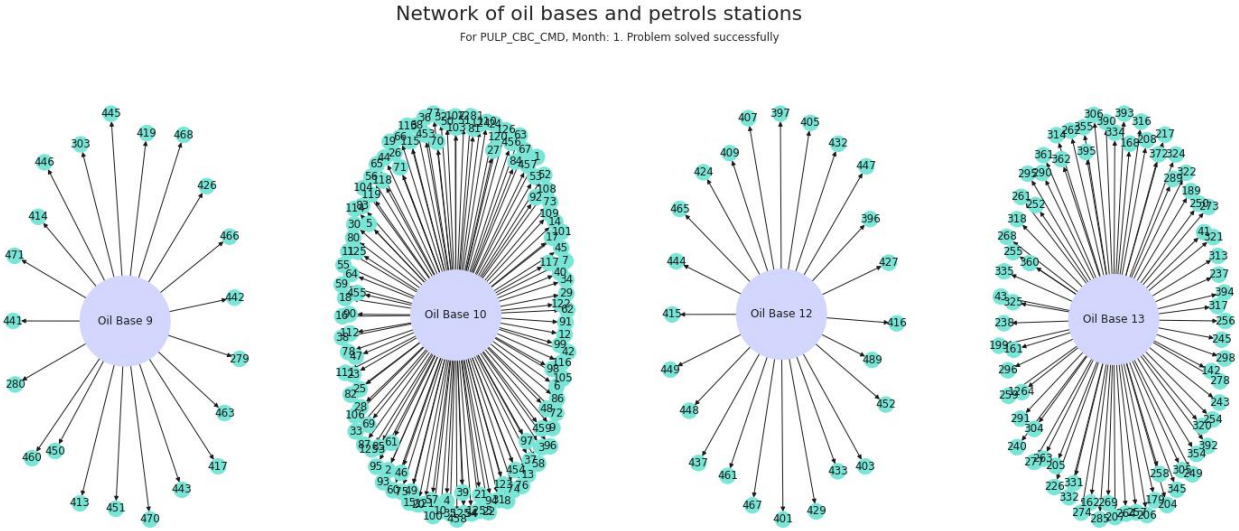


Figure 15. Network of oil bases and petrol stations for current solution for April. Source: (Provided by authors, 2023)

Figure 16 below reflects the costs distributed by brands for CBC solver. Here and on the next figure there are brands that correspond to the numbers of brands from Appendix 2. More detailed information on these brands was given in Table 3.

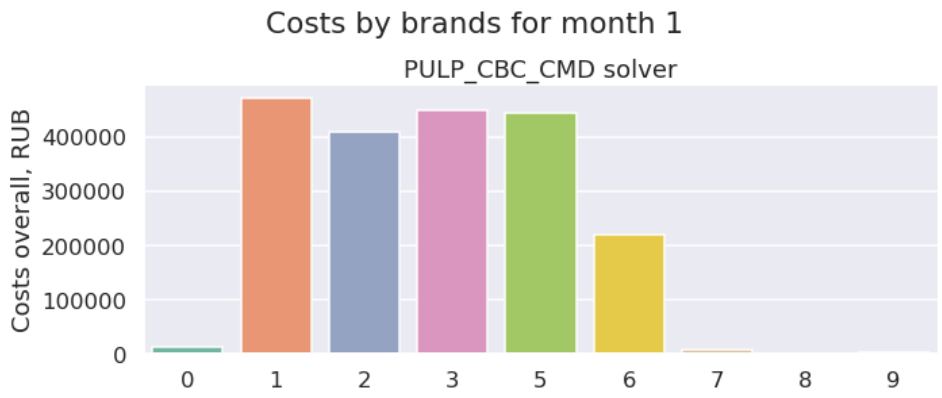


Figure 16. Example of distribution of costs per brands in current solutions for the first month. Source: (Provided by authors, 2023)

Figure 17 below reflects the volumes in tons distributed by brands for CBC solver. Those brands for which volumes were not distributed for the first month were excluded from Figure 16 and Figure 17 for the simplification.

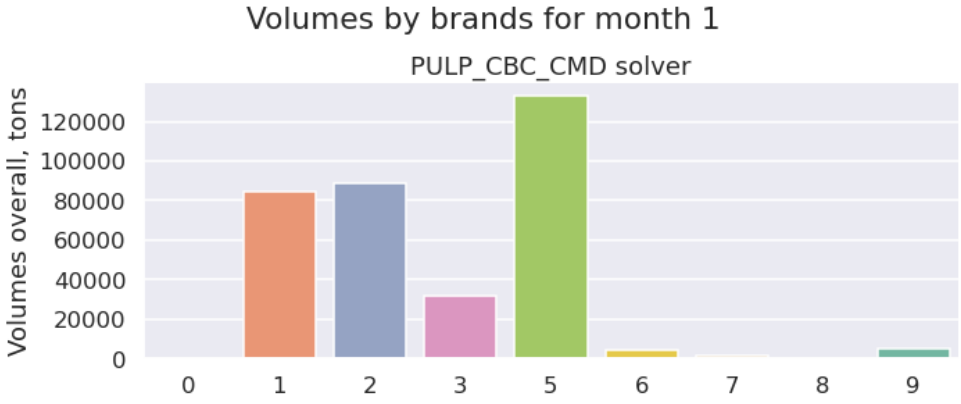


Figure 17. Example of distribution of volumes per brands in current solutions for the first month. Source: (Provided by authors, 2023)

3.2.3. Sensitivity analysis

Figure 18 below presents an example of sensitivity analysis for current solution for the first month. There are different costs depending on loading thresholds for oil bases when supplies are used up to certain degrees. It is important to mention that these percent loadings are constants that can take the values from zero (0%) to one (100%) in the current figure. It can be noted that around 0.8 the model fails to deliver the solution which is discussed more in detail further.

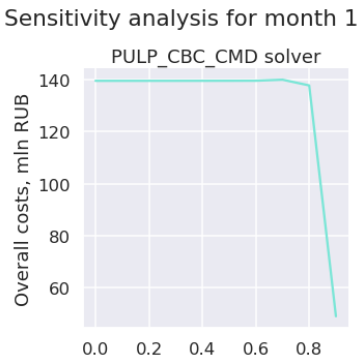


Figure 18. Example of sensitivity analysis for current solution for the first month. Source: (Provided by authors, 2023)

3.3. Discussion on user interaction with model

So, in the end both CBC solver from PuLP library and CBC solver from CVXPY library were chosen as the final solution, but this choice mainly depends on the size of dataset and complexity of the optimization problem. For smaller dataset and not complex problem SCIP solver from PuLP library also could be used because it showed better results in terms of costs on data on three regions, as well as other solvers like COPT from CVXPY library.

It should be noted once again that user can interact with the program and vary configuration file by specifying constants according to some logic. The main attention should be paid to such parameters of configuration file as a set of regions, a set of origins to control and kinds of function for supply upper and lower bound. Some combination of constants may lead to the optimization problem being not solved, so that an optimal solution will not be obtained. Below on Figure 19 action plan is proposed that should help to deal with such cases.

If there is the fact that the problem is not solved, then the following sequence of actions from Figure 19 should be implemented. First of all, a problem is to be defined and a demand constraint is to be added. If the problem is not solved in this case, the first action could be to try another device or environment to work with the model because sometimes execution of solvers depends on technical resources. If the problem is still not solved after this action, then most likely there is an error in the model, and it should be reviewed. If everything works correctly under this restriction, then it is needed to add the next restriction.

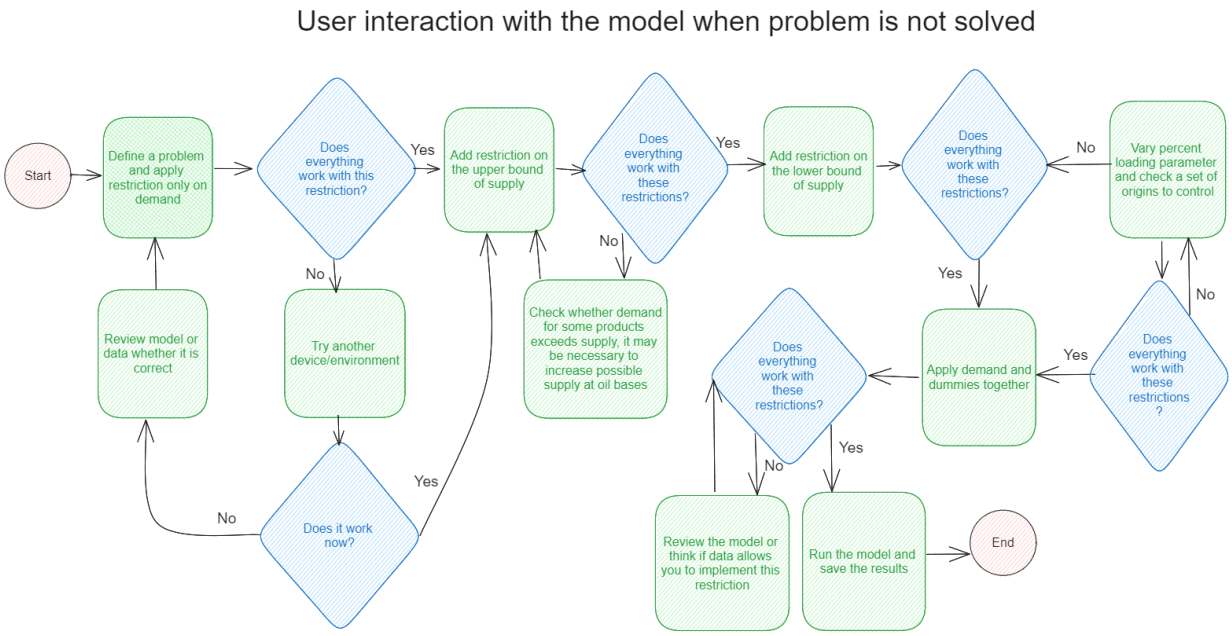


Figure 19. Process of user interaction with model when problem is not solved. Source: (Provided by authors, 2023)

Then the next restriction should be added which includes the upper bound on supply. It is possible to test two versions of this restriction: a softer restriction by product groups and a tougher one by products. If such a combination of demand and upper bound of supply restrictions does not lead to a solution to the problem, then Table 5 previously described probably should be checked. If for some products the demand exceeds the supply, and not vice versa, then apparently the possible supply is not enough to cover the existing demand. This may mean that it is necessary to increase the possible supply at oil bases. It is possible to consider the option of increasing the number of oil bases, as well as connecting oil bases from other regions for those regions where there is not enough supply.

Also, when there is a consideration of the model without restriction on a lower bound of supply, it can be checked which oil bases are not highly loaded or from which products are not being transported. It may be necessary to reorganize such oil bases, maybe it should be thought about closing them or changing the transportation tariffs from these oil bases.

If the problem is solved considering the restrictions on demand and the upper bound of supply mentioned above, then it is necessary to add a restriction on a lower bound of supply. In this case, if the problem is not solved, then there are two parameters that can be varied to correct the situation. It can be seen on Figure 18 based on sensitivity analysis that starting from a certain percent loading for

lower bound of supply, the problem becomes unsolvable. Therefore, it is possible to relax the restriction on the degree of loading of oil base until the problem is solved. In addition, a set of origins to control can be checked. Some oil bases may provide too large supply, which does not satisfy the conditions. The set of origins to control can be revised in terms of exclusion of data on some oil bases. But also, a possible option may be not to include restriction on a lower bound of supply that is shown by the cycle on the flowchart.

Then the restriction related to dummies should be checked together with demand. In the case when the problem is still not solved, then the model should be revised, or data should be checked to see whether it allows to implement this restriction. It should be mentioned that on the graph there is a cycle that means that either this restriction should be eliminated, or the model is adjusted to the data.

Additionally, the case presented in Table 6 may be worth checking. There are some unusual cases where the proportions of supplies to related demands exceed 1. It may be necessary to revise the network of petrol stations in such a way as to expand it and then demand would better match supply.

After all the proposed changes have been made, the situation should be corrected, and the results of the model should be checked again.

3.4. Summary of chapter 3

In this chapter the results yielded with the usage of different libraries and solvers were evaluated based on the criteria and weights.

In Tables 11 and 12 libraries were evaluated based on the following criteria: flexibility, interactivity, quality of documentation. Each library (PuLP, CXVPY, Pyomo, SciPy) used and (or) considered for technical implementation in chapter 2 were given a score from 1 to 5. The important note to make here is that it was decided to avoid using SciPy library because of its low flexibility and a relatively difficult interface; in short, this library utilizes matrixes as the main data format and math formulas in its interface which might be complicated for some users; though SciPy's matrix approach speeds up the calculations, its technical implementation becomes more complicated then more dimensions are added to the data (like product, brands in the case of this paper). The application of weights to the evaluation of libraries is shown in Table 13. Here, CVXPY library was considered to be the best option among four libraries.

In Table 14 the solvers available based on the problem at hand were compared based on their speed and the accuracy of the result they provide. Since solvers rely on different methods (Table 9), the value of an objective function might be different. In Table 15 a combination of separate evaluation of libraries and solvers is obtained. Here the solver that received the highest score was CBC from

PuLP library. Such a combination of solver and library can be an answer to the research question of this paper of which of the existing methods in the field of linear programming and related approaches is mostly applicable to the fuel delivery transportation problem in terms of current business restrictions. It should be noted that in the end CBC solver from CVXPY has demonstrated better results in terms of cost function on the whole dataset, but still the evaluation based on criteria is valid for the test dataset of three regions.

Then the approach to the visualization of model's results was proposed in Python; the visuals are shown in Figures 12-18. The visualization contains: the technical value and the business value of the solution relative to the baseline, the solution itself (graph/map with oil bases and petrol stations and bar charts with costs per product and per brand), and the robustness of the solution (line chart with sensitivity analysis). The user can apply this approach to visualize his or her results.

Finally, discussion on user interaction with model and error handling was provided in Figure 19. It shows the sequence of actions that can be taken by the user when the problem is not solved.

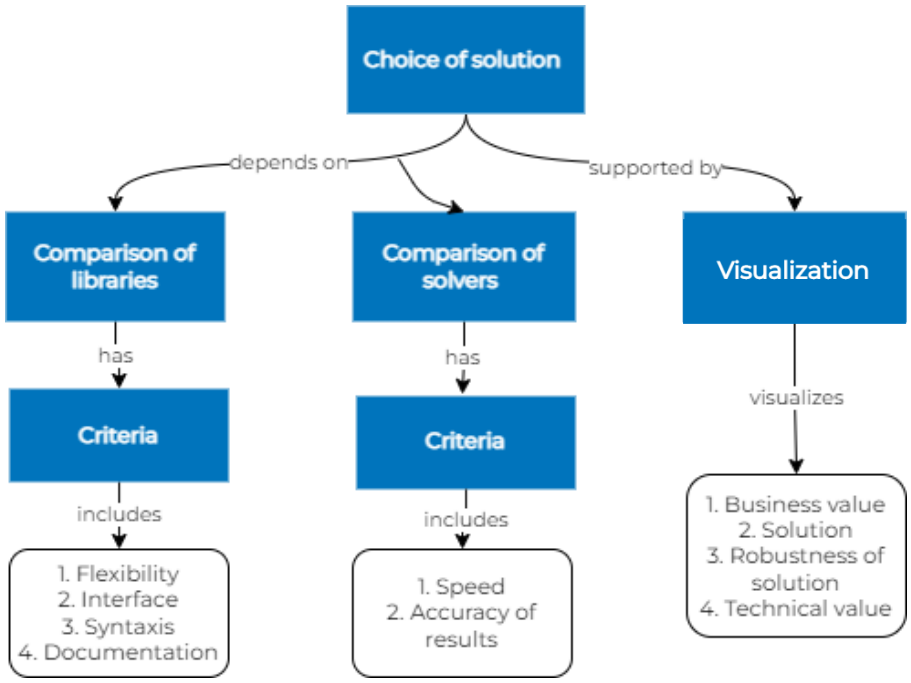


Figure 20. Project plan. Source: (Provided by authors, 2023)

Main logical elements of this chapter and connections between them are present in Figure 20. To sum up, the choice of the final solution depends on the results of the comparisons of libraries and solvers; comparison in each case relies on the criteria. The choice is supported by figures that help to visualize such important aspects as business value, the solution itself, etc.

CONCLUSION

To sum up, the thesis paper was devoted to fuel delivery network optimization for Gazprom Neft company and involved a development of an automated model of fuel transportation from oil bases to petrol stations, considering several business restrictions. The *business goal* of the study, which was to minimize transportation costs (OPEX) for the Gazprom Neft delivery network, was successfully fulfilled. To achieve this, certain objectives listed below were accomplished. Also, an answer was received to a research question related to the study of which of the existing methods in the linear programming area and related approaches is mostly applicable to the fuel delivery transportation problem in terms of current business restrictions. Such a combination of solver and library is currently CBC from PuLP library based on the Branch and Cut method (Table 15) — *research goal* is achieved. All in all, the *goal of the thesis* — to determine the most suitable methods and tools given existing business requirements for minimization of transportation costs for delivery network of Gazprom Neft company — was successfully completed.

In the first part of the paper, based on the overview of the theoretical background, optimization problems were investigated, to which transportation problems may belong and their taxonomy was compiled (Figure 3). Then the methods for solving them were summarized (Figure 4), and a mind map of libraries and solvers was created (Figure 5), from which the appropriate ones can be chosen based on the current type of problem.

In the second chapter, the description of secondary data provided by Gazprom Neft company was given (Table 2). Results of data exploration and summarization were provided in Tables 3-6 and in Figures 7 and 8. Then, the mathematical model of the current transportation problem with constraints was formulated (formulas 15-22). Also, some variations of the mathematical model were presented in formulas 23-25. It was found out that this optimization problem refers to the Mixed Integer Linear Problem, or MILP, because the model contains continuous and discrete variables (Figure 9). Suitable libraries and solvers were selected for this type of problem based on the approach developed in the theoretical part (Figure 10). As the model is designed to be flexible, in Table 7 the configuration of the model was described: the user was able to choose origins to set the lower bound of supply to, both kind of lower and upper bounds for supply, list of regions and date. A few experiments were conducted: variations of the mathematical model were tested in Table 8; all the solvers were tested on a part of the dataset, with the results present in Table 9; it should be mentioned that not all the solvers from Figure 10 could be successfully implemented; for more details, refer to Table 10. At the end of the second chapter, the final solution was given; the whole dataset was tested on CBC solvers from

PuLP and CVXPY libraries; both of these tools provided the result with the costs lower than the baseline specified by the company, and the time of performance close to the required time limit of 25 minutes.

In the last chapter, a set of criteria was developed to evaluate the results yielded with the usage of different libraries and solvers. Libraries were compared in Tables 11-15 based on criteria such as flexibility, interactivity, quality of documentation; and solvers — based on the speed and accuracy of the results. In order to combine different criteria, a set of weights was applied. All the criteria were discussed with the company's representatives. As a result, the best libraries (CVXPY, PuLP) and solvers for solving the MILP transportation problem (CBC from PuLP, CVXPY) were selected. In addition, a visualization in Python of such important aspects as business value, the solution itself, etc. was made. In Figures 12-18 the possible approach to the visualization of model's results that one can use is presented. Finally, the discussion was performed, where the details on user interaction with model were described in the form of a flowchart (Figure 19) with possible options for error handling. This flowchart can be used for debugging.

As for the managerial implications of the results of this study, the resulting model of fuel transportation from oil bases to petrol stations, which is more advanced than the previously existing one, can be applied at the FLD (Fuel and Logistics Department) level of Gazprom Neft company. This model meets business requirements and can automate the process, as well as save time and labor costs. Thus, the provided solution, which is focused on minimizing operational costs (OPEX) for logistics, may help to guarantee the mission of the department to minimize the cost of oil products at salespoint by timely fulfilling the needs to the full extent, while maintaining the product quality. It should be noted that different solutions can be made based on the model's assumptions; moreover, the model can highlight some areas for development inside the company. For instance, if no lower bound is specified and the model underloads a particular oil base, it might be the sign of managerial issue (high tariffs). In addition, exploratory analysis performed in this paper revealed that some oil bases have more supply when they can potentially deliver to petrol stations (see Table 6); this might be the sign that for such oil bases the network of transportations could be expanded. Finally, the overview of tools and methods provided in this paper can be used by those who have to solve similar tasks (for example, in the retail and transport sector) and have various professional backgrounds.

It is worth mentioning that this study has certain limitations. First of all, not all possible solvers, which were discussed in methodology, could be installed due to technical limitations and their commercial nature, in the future more attention could be paid to them. Additionally, in this study, only

the quantitative research method was used, which consists of formulation of a mathematical model, and then creation and implementation of an automated model for optimization in Python. Adding the qualitative methods presented below could overcome this limitation. Another serious limitation of this paper is that some solvers have demonstrated inconsistent behavior, some were excluded based on their inability to deal with big number of constraints.

As for further research, there is a direction within which it would be possible to enhance the methodological part of the research and to overcome the limitations of this research: it could be possible to conduct interviews with representatives from several oil and gas companies to collect current solutions for optimization of fuel delivery. Additional research question may be raised: «What methods are already applied in the industry to solve current optimization problem? » The order of methods in this option would be sequential: initially, a qualitative study is conducted based on interviews conducted, a pool of methods from several companies is analyzed and compared based on certain criteria. Then a quantitative study should be carried out, presented in this paper. This approach would enhance the possible choice of methods and strategies based on the experience of other companies.

All in all, the goal of the current study was achieved, and an automated model was developed and applied to minimize transportation costs (OPEX) for the delivery network of Gazprom Neft company.

LIST OF REFERENCES

Bazaraa, M. S., Sherali, H. D., & Shetty, C. M. (2013). Nonlinear programming: theory and algorithms. *John Wiley & Sons, New York.*

Bertsekas, D. P. (1982). Constrained Optimization and Lagrange Multiplier Methods. *New York: Academic Press.*

Birolini, S., Jacquillat, A., Cattaneo, M., & Antunes, A. P. (2021). Airline Network Planning: Mixed-integer non-convex optimization with demand–supply interactions. *Transportation Research Part B: Methodological*, 154, 100-124.

Charnes, W., Cooper, W., & Henderson, A., (1953). An Introduction to Linear Programming. *John Wiley & Sons, New York.*

CVXOPT. (n.d.). https://cvxopt.org/

CVXPY. (n.d.). https://www.cvxpy.org/

Dantzig, G., (1956). The Simplex Method. *Santa Monica, CA: RAND Corporation.* https://www.rand.org/pubs/papers/P891.html

De Santis, M., Eichfelder, G., Niebling, J., & Rocktäschel, S. (2020). Solving multiobjective mixed integer convex optimization problems. *SIAM Journal on Optimization*, 30(4), 3122-3145.

Diamond, S., & Boyd, S. (2016). CVXPY: A Python-embedded modeling language for convex optimization. *The Journal of Machine Learning Research*, 17(1), 2909-2913.

Diwekar, U. M. (2020). Introduction to applied optimization. *Springer Nature*, 22.

Ford, L. R., & Fulkerson, D. R. (1956). Solving the Transportation Problem. *Management Science*, *3*(1), 24–32.

Forrest, J., & Lougee-Heimer, R. (2005). CBC user guide. *In Emerging theory, methods, and applications*, 257-277.

Gambella, C., Ghaddar, B., & Naoum-Sawaya, J. (2021). Optimization problems for machine learning: A survey. *European Journal of Operational Research*, 290(3), 807-828.

Gass, S. I. (2003). Linear programming: methods and applications (5th ed.). *Dover Publications*.

Gomory, R. (1958). Outline of an Algorithm for Integer Solutions to Linear Programs. *Bulletin of the American Mathematical Society*, 64, 275-278.

Hart, W. E., Laird, C. D., Watson, J., Woodruff, D. L., Hackebeil, G. A., Nicholson, B. L., & Siirola, J. D. (2017). Pyomo — Optimization Modeling in Python. *Springer*.

Hiriart-Urruty, J. B., & Lemaréchal, C. (2004). Fundamentals of convex analysis. *Springer Science & Business Media.*

Hossain, M. M., & Ahmed, M. M. (2020). A comparative study of initial basic feasible solution by a least cost mean method (lcmm) of transportation problem. *American journal of operations research*, 10(04), 122.

Jeyakumar, V., & Rubinov, A. (2006). Continuous Optimization: Current Trends and Modern Applications. *Springer Science & Business Media*. ISBN 978-0-387-26771-5.

Kantorovich, L. (1942) On the translocation of masses. *Dokl. Akad. Nauk SSSR, 37:*199-201.

Karlof, J. K. (2005). Integer programming: theory and practice (1st ed.). *CRC Press.*

Karmarkar, N. (1984). A new polynomial-time algorithm for linear programming. *Proceedings of the sixteenth annual ACM symposium on Theory of computing*, 302-311.

Land, A. H., & Doig, A. G. (1960). An Automatic Method of Solving Discrete Programming Problems. *Econometrica*, 28(3), 497-520.

Lee, J., & Leyffer, S. (2011). Mixed integer nonlinear programming. *Springer Science & Business Media*, 154.

Levitin, E. S., & Polyak, B. T. (1966). Constrained minimization methods. *USSR Computational mathematics and mathematical physics*, 6(5), 1-50.

Meindl, B., & Templ, M. (2012). Analysis of commercial and free and open source solvers for linear optimization problems. *Eurostat and Statistics Netherlands within the project ESSnet on common tools and harmonised methodology for SDC in the ESS*, 20.

Mitchell, J. E. (2002). Branch-and-cut algorithms for combinatorial optimization problems. *Handbook of applied optimization*, 1(1), 65-77.

Monge, G. (1781). Dissertation on the theory of cuttings and embankments. History of the Royal Academy of Sciences of Paris, with the Memoirs of Mathematics and Physics for the same year, 666-704.

MOSEK. (n.d.). https://www.mosek.com/

Nelder, J. A., & Mead, R. (1965). A simplex method for function minimization. *The computer journal*, 7(4), 308-313.

Nemhauser, G. L., Savelsbergh, M. W., & Sigismondi, G. C. (1994). MINTO, a mixed INTeger optimizer. *Operations Research Letters*, 15(1), 47-58.

PuLP. (2022, November 3). PyPI. https://pypi.org/

Pulp Documentation. (2023, March 27). Pulp Project. https://docs.pulpproject.org/pulpcore/#

Pyomo. (n.d.). Pyomo. https://www.pyomo.org/

Pyomo Documentation. (2023, April 4). Readthedocs. https://readthedocs.org/

Ranasinghe, R. M., & Rathnayaka, R. M. K. T. (2021). North West Corner Rule based programming development to find the initial basic feasible solution of transportation problem.

Reeb, J. E., & Leavengood, S. A. (2002). Transportation problem: a special case for linear programming problems.

SCIP. (n.d.). Www.scipopt.org. Retrieved April 15, 2023, from https://www.scipopt.org/

SciPy. (n.d.). https://scipy.org/

Schrijver, A. (2003). Combinatorial Optimization: Polyhedra and Efficiency. *Algorithms and Combinatorics*, 24.

Trevisan, L. (2011). Combinatorial optimization: exact and approximate algorithms. *Stanford University*.

Uddin, M., Khan, A., Kibria, C., & Raeva, I. (2016). Improved Least Cost Method to Obtain a Better IBFS to the Transportation Problem. *Journal of Applied Mathematics and Bioinformatics*, 6, 1-20.

Vigerske, S., & Gleixner, A. (2018). SCIP: Global optimization of mixed-integer nonlinear programs in a branch-and-cut framework. *Optimization Methods and Software*, 33(3), 563-593.

Wulandari, D. A. R., Arifin, F. N., & Santika, G. D. (2018). Implementation North West Corner and Stepping Stone Methods for Solving Logistical Distribution Problem in Tape Production. *In 2018 2nd East Indonesia Conference on Computer and Information Technology* (EIConCIT), 133-138.

APPENDIX

Appendix 1. Allocation of work

		Anastasiia Panchenko	Anna Semenova	Elizaveta Sivash
Literature review	Problems			
	Methods			
	Tools			
Methodology	PuLP			
	CVXPY			
	Pyomo			
Results	Criteria			
	Visualization			

Appendix 2. Indexes of brands

Brand number	Name of brand
0	Gasoline 100 brand
1	Gasoline 92
2	Gasoline 95
3	Gasoline 95 brand
4	Summer diesel fuel with additives
5	Winter diesel fuel with additives class 2
6	Gasoline 100 brand with additives
7	Winter diesel fuel brand class 2
8	Winter diesel fuel brand class 2
9	Diesel fuel with additives demi-season sort F
10	Gasoline 92 optimum
11	Gasoline 95 optimum
12	Diesel fuel Artic class 4
13	Diesel fuel Artic with additives class 4
14	Diesel fuel demi-season brand sort F
15	Diesel fuel demi-season sort F
16	Summer diesel fuel brand
17	Summer diesel fuel

Appendix 3. Technical implementation of the model in Python

The repository with the code and source data is located in GitHub by the following link: https://github.com/annasemenova15/Gazpromneft_project