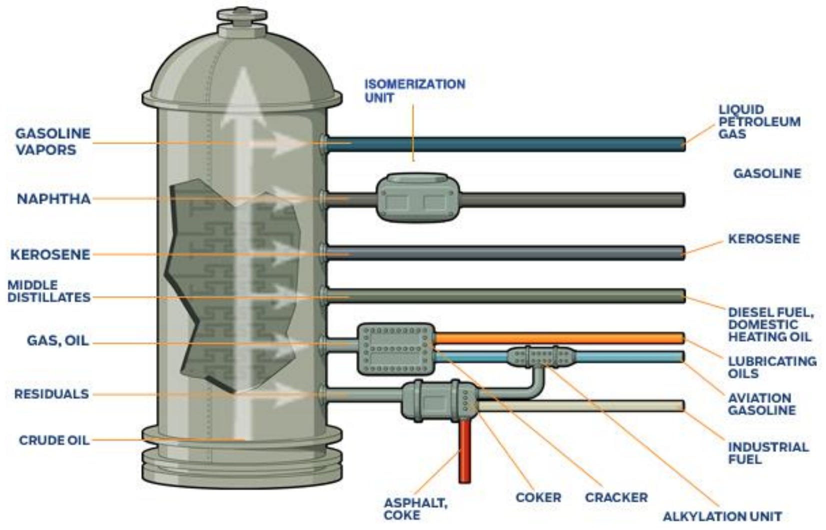


POOLING PROBLEM IN PETROLEUM INDUSTRY

DATA STRUCTURES AND ALGORITHMS

SAURABH GUPTA (72078)
B.TECH. (INFORMATION TECHNOLOGY AND MATHEMATICAL INNOVATIONS)

ABSTRACT

The pooling problem is an important optimization problem that is encountered in operation and scheduling of important industrial processes within petroleum refineries. The key objective of pooling is to mix various intermediate products to achieve desired properties and quantities of products. First, intermediate streams from various processing units are mixed and stored in intermediate tanks referred to as *pools*. The stored streams in pools are subsequently allowed to mix to meet varying market demands. While these pools enhance the operational flexibility of the process, they complicate the decision making process needed for optimization. The problem to find the least costly mixing recipe from intermediate streams to pools and then from pools to sale products is referred to as the *pooling* problem. The research objective is to contribute an approach to solve this problem.

Keeping the optimization problem in mind a model is developed which is flexible enough to be implemented in different real scenarios. Computationally, the model is implemented using PuLP library in python. Basically, the problem is divided into two parts, one of them is minimizing problem and other one is maximizing problem. The final aim is to maximize the final profit to the refinery by minimizing the cost of production and maximizing the selling price of the final products.

The model can help a refinery to optimize their functionality in order to maximize their profit keeping in mind to retain the quality of the products.

ACKNOWLEDGEMENT

I have taken efforts in this project. However, it would not have been possible without the kind support and help of many individuals and organizations. I would like to extend my sincere thanks to all of them.

I am highly indebted to Mr. Alok Nikhil Jha for his guidance and constant supervision as well as for providing necessary information regarding the project & also for their support in completing the project.

I would like to express my gratitude towards my parents & members of Cluster innovation Centre for their kind co-operation and encouragement which help me in completion of this project.

Thank You,

Saurabh Gupta

1 CONTENTS

2 INTRODUCTION

During operation and scheduling of important industrial processes inside a petroleum refinery, there occur some optimization problems that are known as pooling problems. Pooling is done to achieve some specific desired output quantities and qualities of products by mixing various intermediate products.

Basically, pools are the intermediate tanks in which intermediate streams from various processing units are mixed. And finally, the blend present in the pools is further allowed to mix and form a final product to meet the various market demands.

Due to these pools, the operational flexibility of the petroleum refining process is increased but the pools makes it difficult to make decisions regarding quantities of flows and hence need optimization.

The problem is to optimize:

- Minimize the cost of production of different products,
- Maximize the selling price of the final products,

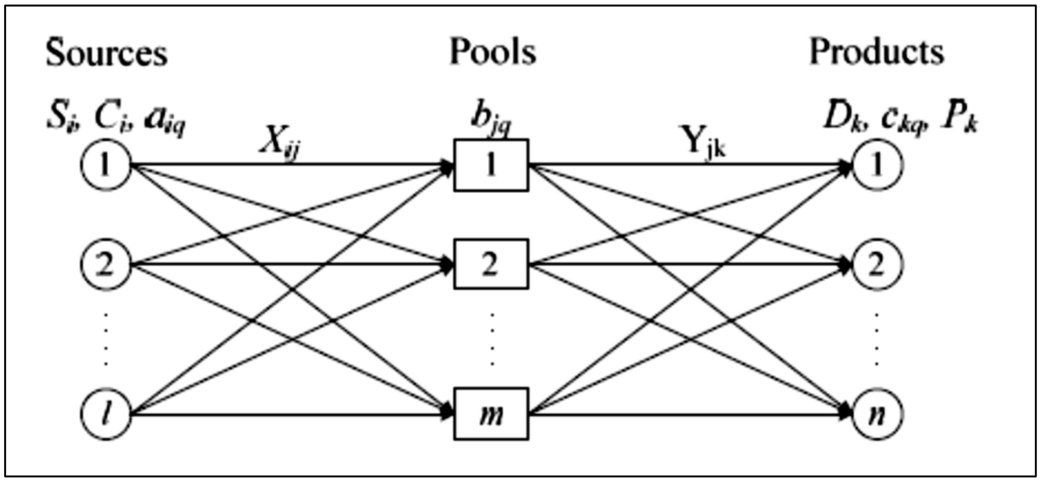
in order to maximize the profit.

All these are non-convex optimization problems.

3 FORMING A PROBLEM MODEL

Let

- i represent Sources
- j represents pools
- k represents products
- q represents quality
- S_i represents available capacity of source i
- D_k represents demand of product k
- a_i represents source i quality
- c_k represents product k quality
- P_k represents price of product k
- C_i represents cost of source i
- X_{ij} represents amount of flow from source i to pool j
- Y_{jk} represents amount of flow from pool j to product k



Due to varying market demands, we assume that there are '*l*' sources, 'm' number of pools and 'n' number of products. So, the amount of flows of petroleum and the processed petroleum between source and pools and pools to final products respectively is assumed to be random.

With the above notions, the problem is formulated as:

3.1 OBJECTIVE FUNCTION:

The objective of any petroleum refinery is to earn most of the profit. To do so, they need to keep two things in mind, one is to minimize the cost of production and the other is to maximize the selling price, and thereby, maximize the profit. So, our objective function goes as:

$$\sum_{k=1}^{n} P_k \cdot \sum_{j=1}^{m} Y_{jk} - \sum_{i=1}^{l} C_i \sum_{j=1}^{m} X_{ij}$$

3.2 CONSTRAINTS:

- **Available Supply:**

  The flows 'X_{ij}' from a source 'i' cannot exceed available capacity 'S_i' of the source. So, the constraint will be as:

$$\sum_{j=1}^{m} X_{ij} \leq S_i \qquad for\ i = 1,2, \ldots., l$$

- **Mass balance on Pools:**

  The amount of flows 'X_{ij}' from the source 'i' to the pool 'j' and then 'Y_{jk}' from the pool 'j' to the final product 'k' must be equal. So, the constraint will be as:

$$\sum_{i=1}^{l} X_{ij} = \sum_{k=1}^{n} Y_{jk} \qquad for\ j = 1,2, \ldots., m$$

- **Product Demand:**

  According to the varying market, the flows 'Y_{jk}' coming in to the products 'k' must be less than or equal to the market demand 'D_k' of that product. So, the constraint will be as:

$$\sum_{j=1}^{m} Y_{jk} \leq D_k \qquad for\ k = 1,2, \ldots., n$$

- **Mixing Rule for Pools:**

  The quality 'b_{jq}' of flows 'X_{ij}' to the pools must be greater than the quality 'a_i' in the sources. So, the constraint will be as:

$$b_{jq} \sum_{i=1}^{l} X_{ij} \geq \sum_{i=1}^{l} X_{ij}\, a_{iq} \qquad for\ j = 1,2, \ldots., m$$

- **Mixing Rules for Products:**
  The quality 'c_{kq}' of products must be greater than the quality 'b_{jq}' of the flows 'Y_{jk}' from the pools. So, the constraint will be as:

$$c_{kq} \sum_{j=1}^{m} Y_{jk} \geq \sum_{j=1}^{m} Y_{jk}\, b_{jq} \qquad\qquad for\ k = 1, 2, \ldots., n$$

- **Non-negativity constraints:**
  The flows 'X_{ij}' from the sources to the pools and 'Y_{jk}' from the pools to the final products cannot be negative. So, the constraint will be as:

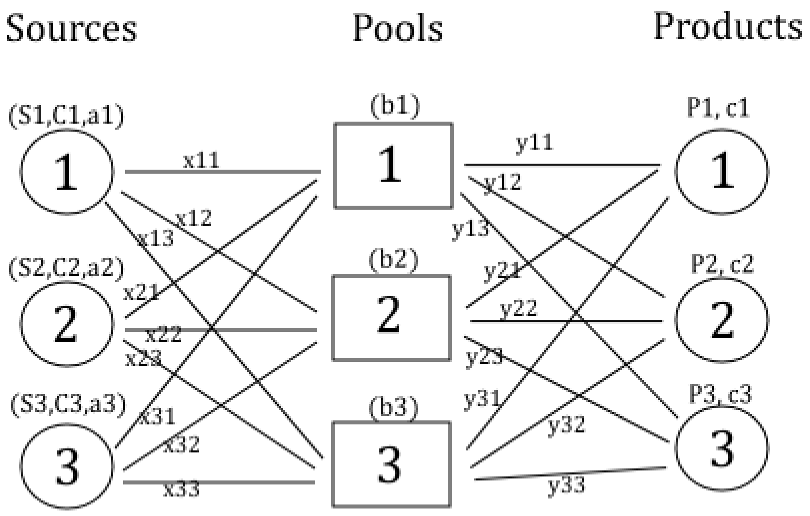
$$X_{ij} \geq 0 \qquad\qquad for\ i = 1, 2, \ldots., l\ \&\ j = 1, 2, \ldots., m$$
$$Y_{jk} \geq 0 \qquad\qquad for\ j = 1, 2, \ldots., m\ \&\ k = 1, 2, \ldots., n$$

4 APPROXIMATIONS

Before doing anything we have to find the approximate of anything regarding the problem so that the proposed model has some relevance to reality. Otherwise, generating random values within any range doesn't make any sense. So, some approximations are to be done so as to find the range of the random values to be given to the known identities within the problem.

- The source of our problem is different types of petroleum. So, the source is same, neglecting quality constraints, we take the range of 'S_i' [90, 100).
- The qualities 'a_{iq}', 'b_{jq}' and 'c_{kq}' are assumed to be within (0, 1).
- The demand of the product varies according to the market. So, the demands 'D_k' are taken to be within [90, 100).
- The most common products of petroleum refineries are gasoline, jet fuel, and diesel. So, their market prices 'P_k's (Mumbai) are taken to be within [70, 80), [35, 45) and [55, 65) respectively.
- And according to some research the cost of production is approximately Rs.20 less than the market price. So, the costs of production 'C_i's are taken to be [50, 60), [15, 25), [35, 45) respectively.

5 AN EXAMPLE (3 SOURCES, 3 POOLS & 3 PRODUCTS)



Let,

- S1, S2, S3 are available supply capacities at sources 1, 2 & 3 respectively.
- C1, C2, C3 are cost of production of sources 1, 2 & 3 respectively.
- a1, a2, & a3 are quality of flows from sources 1, 2 & 3 respectively.
- {X11, X12, X13} are flows from source 1, {X21, X22, X23} are flows from source 2, {X31, X32, X33} are flows from source 3 to pools 1, 2 &3 respectively.
- b1, b2 & b3 are quality at the pools 1, 2 & 3 respectively.
- P1, P2 & P3 are prices of the products 1, 2 & 3 respectively.
- c1, c2, &c3 are quality of the final products 1, 2 & 3 respectively.
- {Y11, Y12, Y13} are flows from pool 1, {Y21, Y22, Y23} are flows from pool 2, {Y31, Y32, Y33} are flows from pool3 to final products 1, 2 & 3 respectively.

5.1 OBJECTIVE FUNCTION

We have to maximize:

$$P1(Y11+Y21+Y31)+P2(Y12+Y22+Y32)+P3(Y13+Y23+Y33) - C1(X11+X12+X13)-C2(X21+X22+X23)-C3(X31+X32+X33)$$

5.2 CONSTRAINTS

- **Available Supply:**

$$Source\ 1:\quad X11+X12+X13 \le S1$$
$$Source\ 2:\quad X21+X22+X23 \le S2$$
$$Source\ 3:\quad X31+X32+X33 \le S3$$

- **Mass Balance on pools:**

$$Pool1:\quad Y11+Y12+Y13 = X11+X21+X31$$
$$Pool2:\quad Y21+Y22+Y23 = X12+X22+X32$$
$$Pool3:\quad Y13+Y32+Y33 = X13+X23+X33$$

- **Product Demand:**

$$Product\ 1:\quad Y11+Y21+Y31 \le D1$$
$$Product\ 2:\quad Y12+Y22+Y32 \le D2$$
$$Product\ 3:\quad Y13+Y23+Y33 \le D3$$

- **Mixing Rule for pools:**

$$Pool\ 1:\quad b1(X11+X21+X31) \ge a1X11+a2X21+a3X31$$
$$Pool\ 2:\quad b2(X12+X22+X32) \ge a1X12+a2X22+a3X32$$
$$Pool\ 3:\quad b3(X13+X23+X33) \ge a1X13+a2X23+a3X33$$

- **Mixing Rule for products:**

$$Product\ 1:\quad q1(Y11+Y21+Y31) \ge b1Y11+b2Y21+b3Y31$$
$$Product\ 2:\quad q2(Y12+Y22+Y32) \ge b1Y12+b2Y22+b3Y32$$
$$Product\ 3:\quad q3(Y13+Y23+Y33) \ge b1Y13+b2Y23+b3Y33$$

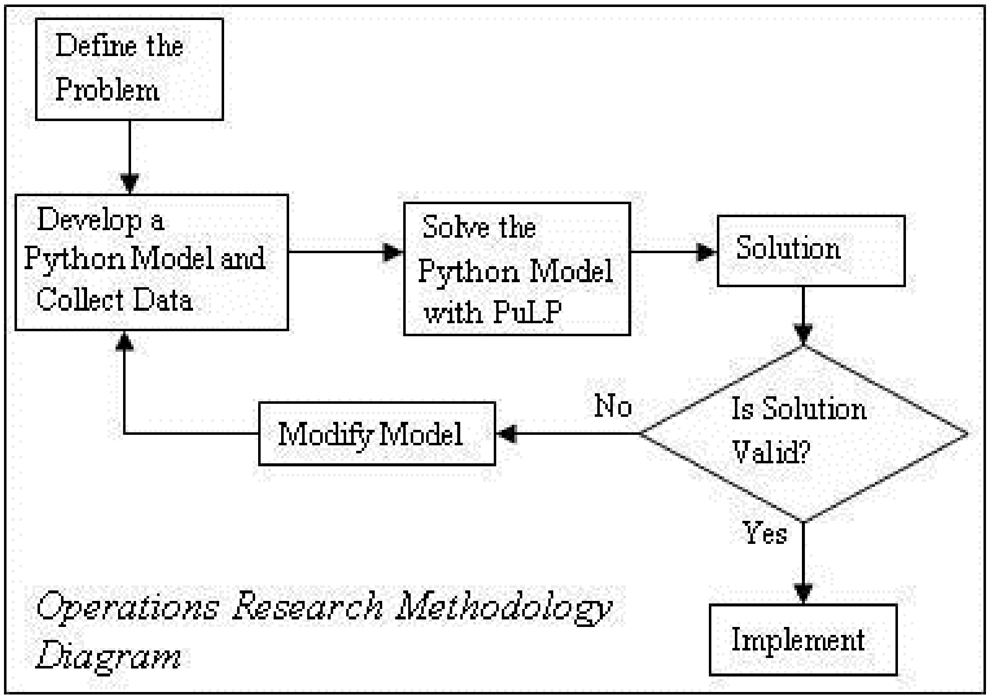
- **Non-negativity constraints:**

$$X11 \geq 0,$$
$$X12 \geq 0,$$
$$X13 \geq 0,$$
$$X21 \geq 0,$$
$$X22 \geq 0,$$
$$X23 \geq 0,$$
$$X31 \geq 0,$$
$$X32 \geq 0,$$
$$X33 \geq 0,$$
$$Y11 \geq 0,$$
$$Y12 \geq 0,$$
$$Y13 \geq 0,$$
$$Y21 \geq 0,$$
$$Y22 \geq 0,$$
$$Y23 \geq 0,$$
$$Y31 \geq 0,$$
$$Y32 \geq 0,$$
$$Y33 \geq 0$$

6 SOLUTION OF THE MODEL

The process of solving a linear programming problem can be divided into five general steps:

1. Getting the problem description.
2. Formulating the mathematical program.
3. Solving the mathematical program.
4. Performing post-optimal analysis.
5. Presenting the solution and analysis.


Define the Problem
Develop a Python Model and Collect Data
Solve the Python Model with PuLP
Solution
Is Solution Valid?
No
Modify Model
Yes
Implement
Operations Research Methodology Diagram

7 IMPLEMENTATION OF THE MODEL

I've used python programming language to implement the proposed model. Basically, PuLP library is used to implement the optimization problem. PuLP is an LP modeler written in python. PuLP can generate MPS or LP files and call GLPK, COIN CLP/CBC, CPLEX and GUROBI to solve linear problems.

7.1 INSTALLING PULP

First you must have a working python installation before installing PuLP. Requires Python version >= 2.5. It is present at https://pypi.python.org/pypi/PuLP/1.5.6 and can easily be downloaded using EasyInstall on windows system and using pip on linux.

```
sudo pip install pulp
```

7.2 GENERATING VARIABLES

We can generate continuous or integer variables. As per our model, our all variables are going to be integers. So, variables can be declared as:

```
X=LpVariable("name", lowerBound, upperBound, LpInteger)
```

7.3 INITIALIZING THE PROBLEM

To form the problem, we first tell whether to minimize or to maximize.

```
Problem=LpProblem("name of the problem", LpOptimize)
```

There are two variables, LpMinimize=1 and LpMaximize=-1 that take place of LpOptimize as per our requirement.

7.4 ADDING THE OBJECTIVE FUNCTION AND CONSTRAINTS

'+=' operator is used to add objective function and contraints to the problem variable (Problem, in this case).

```
Problem+= C1*X11+C2*X12+C3*X13,"Solution to the problem"
```

To add the constraints again we use the same += operator.

```
Problem+= X11+X12+X13>=S1,"name of the constraint"
```

7.5 WRITING THE LP PROBLEM

It is important to write the problem to match and verify it afterwards.

```
Problem.writeLP("with_whatever_name_you_want.lp")
```

7.6 SOLVING THE PROBLEM

To solve the problem, we've to call a pre-defined function in PuLP. This is the most important thing to be done in LP.

```
Problem.solve()
```

7.7 PRINTING STATUS, LP VARIABLE's VALUES, FINAL VALUE (OPTIMIZED)

A solution could be Infeasible, Feasible, bounded, unbounded and Optimal.

```
Print LpStatus[Problem.status]
```

To print values of the LpVariables that are set while defining objective function and constraints.

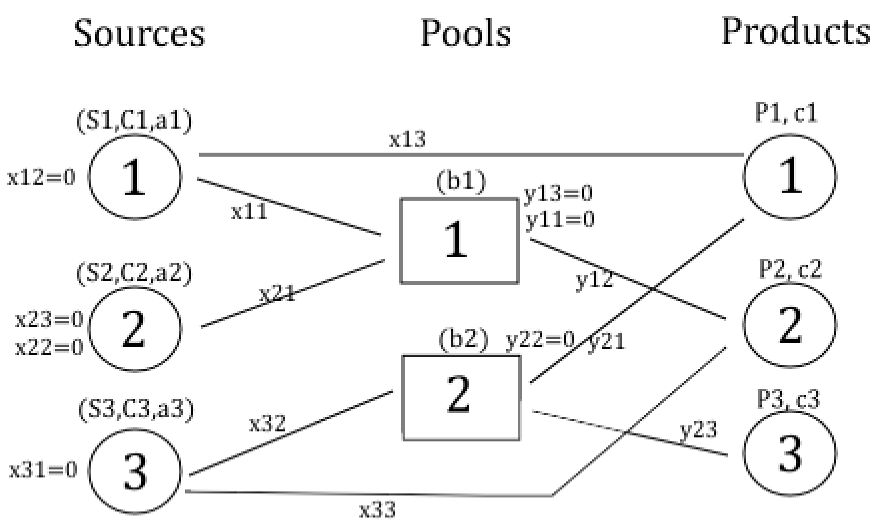
```
for v in Problem.variables():

    print v.name,"=", v.varValue
```

Final Optimized value for the variable

```
print "cost of flow :  ", value(Problem.objective), '\n'
```

Further details and documentation of PuLP can be found on https://www.coin-or.org/PuLP.

8 A DIFFERENT APPROACH



Deviations from the previous model:

1. The values of some flows, X12, X23, X22, X31, Y11, Y13, Y22, Y31 & Y33 are 0.
2. Actually, there is no pool 3. So, we assume pool 3 to be just a channel with quality and cost equal to 0.
3. It can be seen that some flows X13 & X33 are direct from sources to products.

Considering these deviations, the above model can be fitted in the previously prepared model and an optimal solution can be found.

9 CONCLUSIONS

A model with approximate values of all the variables is prepared which takes in values very randomly and generates the status of the problem whether it is bounded, unbounded, feasible, infeasible or optimal. Every time, the model generates an optimal solution to all parts of the problem, the same can be implemented in any real time scenario to maximize the profit, keeping the values of variables like available capacity, demand, flows from sources to pools and pools to products, qualities etc approximately similar to the random values that are generated by the model.

Moreover, the model is too flexible to be implemented in a large number of scenarios by adding or removing the constraints as per the refinery for which optimal solution is to be found.

10 BIBLIOGRAPHY

[1] "http://www.petroleum.co.uk/composition," [Online].

[2] "http://en.wikipedia.org/wiki/Petroleum," [Online].

[3] "http://www.numbeo.com/gas-prices/country_result.jsp?country=India," [Online].

[4] "http://www.indexmundi.com/commodities/?commodity=jet-fuel&months=12&currency=inr," [Online].

[5] "https://www.chem.tamu.edu/class/majors/tutorialnotefiles/percentcomp.htm," [Online].

[6] "http://www.epa.gov/athens/learn2model/part-two/onsite/es.html," [Online].

[7] V. PHAM, "A GLOBAL OPTIMIZATION APPROACH," p. 136, August 2007.

[8] "http://www.coin-or.org/PuLP/," [Online].

11 APPENDIX

- For source code of general pooling optimization problem (3 sources, 3 pools, 3 products) with all constraints, visit:
  https://github.com/Saurabh3012/Optimization-Problems/blob/master/opt_with_all_constraints.py

- For source code of general pooling optimization problem with all constraints with lower bound of all variables as zero, visit:
  https://github.com/Saurabh3012/Optimization-Problems/blob/master/opt_with_lb_zero.py

- For source code of a different approach of pooling problem (3 sources, 2 Pools, 3 products), visit:
  https://github.com/Saurabh3012/Optimization-Problems/blob/master/opt_of_other%20_kind.py

- For source code without quality constraint (adding and removing constraints to verify flexibility of the model), visit:
  https://github.com/Saurabh3012/Optimization-Problems/blob/master/opt_without_quality_constraint.py