# ITERATED INSIDE-OUT :
# a new exact algorithm for the transportation problem

Bargetto R.[1], Della Croce F.[1,2] and Scatamacchia R.[1]

[1] DIGEP, Politecnico di Torino, Corso Duca degli Abruzzi 24, 10129 Torino, Italy.
[roberto.bargetto,federico.dellacroce,rosario.scatamacchia]@polito.it
[2] CNR, IEIIT, Torino, Italy

## 1 Introduction

We consider the well-known transportation problem (TP), one of the paradigmatic network optimization problems present in every operations research textbook. TP can be expressed as follows. There is a given commodity that requires to be shipped from a number of sources to a number of destinations at minimum cost. Let $M$ and $N$ be the set of sources and the set of destinations respectively. Let $a_i$ and $b_j$ denote the level of supply at each source $i \in M$ and the amount of demand at each destination $j \in N$, respectively, where, typically, $\sum_{i \in M} a_i = \sum_{j \in N} b_j$. Let $c_{ij}$ denote the unit transportation cost from source $i \in M$ to destination $j \in N$. Let, also, $x_{ij} \geq 0$ be a real variable representing the quantity sent from source $i \in M$ to destination $j \in N$. A linear programming (LP) formulation of TP reads

$$\min \sum_{i \in M} \sum_{j \in N} c_{ij} x_{ij}$$
$$\sum_{j \in N} x_{ij} = a_i \qquad \forall\, i \in M$$
$$\sum_{i \in M} x_{ij} = b_j \qquad \forall\, j \in N$$
$$x_{ij} \geq 0 \qquad \forall\, i \in M,\, j \in N.$$

Since the work in [4], TP has been intensively studied in the literature. As it can be formulated as a continuous linear program, it can also be solved by the most efficient algorithms available for LP, namely, the primal simplex method, the dual simplex method and the barrier method. Also, the current state of the art algorithm for the minimum cost flow problem (as mentioned in [2]), which generalizes TP, is the so-called network simplex algorithm (knowledge of which is assumed), see e.g. [3], currently available in state-of-the-art mathematical programming solvers. Besides, in [1] a comparison of a series of available softwares for TP (including solvers Cplex and Gurobi) is provided where it appears that approaches based on the simplex algorithm (in its various expressions : primal, dual, network) are the best performing. We propose here a new exact pivot-based approach for TP denoted ITERATED INSIDE-OUT (I-I-O).

## 2 The ITERATED INSIDE-OUT algorithm

The I-I-O algorithm requires an initial basic feasible solution and is split into two phases that are iteratively repeated until an optimal basic feasible solution is reached. In phase 1 (the INSIDE phase), the current basic feasible solution is progressively improved by increasing one after another several non-basic variables with negative reduced cost. This step is performed by progressively updating the value of the starting set of basic variables. We remark that, in phase 1, each variation of a non-basic variable is performed by means of a pivoting step with respect to the original basic solution, corresponding to the search of a path in a tree for TP. At the end of this phase, a non-basic feasible solution is derived which is inside the feasibility region determined by the constraints set. In phase 2, the added variables are considered one at a time and the algorithm progressively moves back (the OUT oriented phase where the solution value is progressively improved or kept equal in case of degeneracy) iteratively reducing the distance from a basic solution. Phase 2 proceeds as indicated until a new basic feasible solution is reached. Also, in this phase, every time a new added variable is considered, a pure pivoting step is necessary (again corresponding to the search of a path in a tree) in order to progressively reduce the number of variables with value superior to 0. Once a new basic solution is obtained, the algorithm recomputes the reduced costs of the non-basic variables and iterates reapplying phase 1. The peculiarity of the proposed approach is that both in phases 1 and 2, differently from a standard simplex iteration that requires the computation of lagrangian multipliers and reduced costs, each pivoting step requires just the computation of a path in a tree, while lagrangian multipliers and reduced costs are computed only once in a while whenever phase 1 restarts. This aspect strongly enhances the performance of the proposed approach. Computational testing is summarized in Table 1 where I-I-O is compared to Cplex and Gurobi (versions 20.1.0.0 and 9.5.1, respectively). All the experiments were run as single thread processes on a laptop personal computer equipped with a *11th Gen Intel Core i7-1165G7 2.80GHz × 8* processor and 16GB of RAM, and running Ubuntu 20.04.5 LTS. All supply/demand quantities were drawn from the discrete uniform distribution $U\{1, 1000\}$. We present here the results on square instances with $K$ sources and $K$ destinations, with $K = 1000, 2000, 4000, 8000$, 16000 and a cost distribution drawn from the discrete uniform distribution $U\{1, K\}$. Ten instances were considered for each value of $K$. We note that I-I-O strongly outperforms all methods listed in the table (that are limited to problems with size up to 4000x4000 - we remark an unusual behavior of Gurobi Primal Simplex on instances 2000x2000) and solves to optimality instances with up to 16000x16000 requiring on the largest instances strictly less than 30 seconds on average. Due to space limitation, we do not describe here the procedure to generate the initial basic solution, but even starting from the well known North-Western Corner rule, 16000x16000 instances are solved within 60 seconds.

| Size | 1000x1000 | 2000x2000 | 4000x4000 | 8000x8000 | 16000x16000 |
|---:|---:|---:|---:|---:|---:|
| Algorithm | CPU (ms) | CPU (ms) | CPU (ms) | CPU (ms) | CPU (ms) |
| Cplex Network Simplex | 1466 | 6923 | 35858 | - | - |
| Cplex Primal Simplex | 1670 | 7546 | 34692 | - | - |
| Cplex Dual Simplex | 2433 | 16351 | 122782 | - | - |
| Cplex Barrier | 7451 | 42969 | 326885 | - | - |
| Gurobi Primal Simplex | 2365 | 216001 | 55684 | - | - |
| Gurobi Dual Simplex | 991 | 3755 | 15834 | - | - |
| Gurobi Barrier | 4508 | 24875 | 142573 | - | - |
| I-I-O | **61** | **229** | **1070** | **4317** | **26320** |

All the CPU times reported in the table are averaged over 10 instances.

TAB. 1 – Comparing I-I-O algorithm to Cplex and Gurobi.

## Acknowledgements

This work has been partially supported by "Ministero dell'Istruzione, dell'Università e della Ricerca" Award "TESUN-83486178370409 finanziamento dipartimenti di eccellenza CAP. 1694 TIT. 232 ART. 6".

## References

[1] Schrieber J., Schuhmacher D., Gottschlich C., "DOTmark - a benchmark for discrete optimal transport", IEEE Access, 5, 271-282, 2016.

[2] Kovács P., "Minimum-cost flow algorithms : an experimental evaluation", Optimization Methods and Software, 30, 94-127, 2015.

[3] Orlin J.B., "A polynomial time primal network simplex algorithm for minimum cost flows", Mathematical Programming, 78 (2), 109-129, 1996.

[4] Hitchcock F.L., "The Distribution of a Product from Several Sources to Numerous Localities", Journal of Mathematics and Physics, 20, 224-230, 1941.