# The multicommodity multilevel bottleneck assignment problem

(Article begins on next page)

16 August 2024

# UNIVERSITÀ DEGLI STUDI DI TORINO

**This is an author version of the contribution published on:**

**The definitive version is available at:**

# The Multicommodity Multilevel Bottleneck Assignment Problem

Roberto Aringhieri [a] Roberto Cordone [a]

[a] *DTI - University of Milan*
*via Bramante 65, Crema 26013 ITALY*

**Abstract**

The Multilevel Bottleneck Assignment Problem is defined on a weighted graph of $L$ levels and consists in finding $L - 1$ complete matchings between contiguous levels, such that the heaviest path formed by the arcs in the matchings has a minimum weight. The problem, introduced by Carraresi and Gallo (1984) to model the rostering of bus drivers in order to achieve an even balance of the workload among the workers, though frequently cited, seems to have never been applied or extended to more general cases. In this paper, we discuss one possible extension, that is the introduction of *multicommodity* aspects to model different classes of workers.

*Key words:* Crew Rostering, Bottleneck Assignment

## 1 Introduction

The Bottleneck Assignment Problem is the search for a complete matching on a weighted bipartite graph, such that the weight of the heaviest edge in the matching is minimum. Its multi-level version is defined on a weighted graph of $L$ levels and consists in finding $L - 1$ complete matchings between contiguous levels, such that the heaviest path formed by the arcs in the matchings has a minimum weight. It was introduced by Carraresi and Gallo (1984) to model the rostering of bus drivers in order to achieve an even balance of the workload among the workers. Their algorithm determines a starting feasible solution by solving a sequence of Bottleneck Assignment Problems on the single levels; then, furtherly improves the solution through a "stabilization" process. The final result, though not necessarily optimal, has a bounded gap with respect to the optimum, and is asymptotically optimal for a large time horizon. Our interest in this problem derives from a similar application: the rostering of workers' shifts for the junk removal company of Crema, in Italy. This practical

case, however, requires a more complex model, because each driver is qualified to perform only a subset of the possible shifts.

Crew Rostering has been a lively field of study over the last decades. Most of the approaches in the literature, however, end up with a Set Partitioning model, whose variables correspond to the feasible sequences of shifts assigned to each worker (Caprara, Toth, Vigo and Fischetti , 1998). Some approaches take into account all variables or a heuristic subset, while the others start with a reduced set of promising variables and apply column generation to introduce other variables only if necessary. Beasley and Cao (1998) proposed a dynamic programming approach. Cappanera and Gallo (2003) give a multi-commodity flow formulation for an airline crew rostering problem, which is strengthened by valid inequalities and solved with a general-purpose MIP solver.

The bottleneck approach by Carraresi and Gallo (1984), though frequently cited, seems to have never been applied or extended to more general cases. Therefore, the main concern of this paper is to suggest one of the possible extension, that is the introduction of *multicommodity* aspect to model different classes of workers. After reporting a $\mathcal{NP}$-completeness proof, we propose a Lagrangean Decomposition (Guignard and Kim , 1987). Since the Lagrangean subproblem is not trivial, this paper introduces upper and lower bounding procedures which integrate the ideas in Carraresi and Gallo (1984) in a more general framework.

## 2 Notation and $\mathcal{NP}$-completeness

Given a time horizon of $L$ days, a weighted level graph $G(V, E)$ of $L$ levels models the structure of the service. Its vertices, partitioned into subsets $V_\ell$ associated to the single days ($V = \cup_{\ell=1}^{L} V_\ell$) correspond to the shifts. Without loss of generality, one can assume the number of shifts per day to be equal to the number of workers $n$ ($V_\ell = \left\{ u_1^\ell, \ldots, u_n^\ell \right\}$ for $\ell = 1, \ldots, L$): dummy shifts can be added to guarantee this result. Each shift $u_i^\ell$ implies a workload $w_i^\ell$. The edges of graph $G$ connect only vertices in contiguous levels. They also model trade union agreements: edge $\left( u_i^\ell, u_j^{\ell+1} \right)$ exists if and only if a worker is allowed to perform shift $i$ in day $\ell$ and shift $j$ the day after. Let $S_i^\ell$ denote the subset of vertices in $V_{\ell+1}$ which are linked to vertex $u_i^\ell$ and $P_i^\ell$ the subset of vertices in $V_\ell$ which are linked to vertex $u_i^{\ell+1}$ . The workers form $K$ classes: there are $n_k$ workers of class $k$ (with $\sum_{k=1}^{K} n_k = n$), and they are able to perform only a specific subset $T_k$ of the shifts in $V$.

**Proposition 1** *It is $\mathcal{NP}$-complete to determine whether the* MMBA *problem admits any feasible solution.*

**PROOF.** Given a *SAT* instance with $n$ variables $x_i$ and $m$ clauses $C_j$, build the following *MMBA* instance. Graph $G(V, E)$ is made up of $L = m + 1$ levels of $2n$ vertices each. The vertices $u_i^L$ and $u_{i+n}^L$ of level $L$ are associated to variable $x_i$ $(i = 1, \ldots, n)$. Vertex $u_1^j$ in each of the other $m = L - 1$ levels is associated to the corresponding clause $C_j$ $(j = 1, \ldots, m)$. In each level from 1 to $m$, the first $n$ vertices ($u_i^j$ for $i = 1, \ldots, n$) are linked to all vertices in the following level, while the last $n$ vertices ($u_{i+n}^j$) are only linked to the last $n$ vertices. There are $K = 2n$ classes of workers, which consist of single workers: class $i$ corresponds to literal $x_i$, class $i + n$ to $\bar{x}_i$. Subset $T_i$ includes all vertices of the graph, apart from the $2n - 2$ vertices of level $L$ which are associated to variables different from $i$ and the clause vertices $u_1^j$ such that clause $C_j$ is not satisfied by literal $x_i$; a corresponding definition holds for subset $T_{i+n}$ and literal $\bar{x}_i$. By construction, the path of worker $i$ ends either in $u_i^L$ or in $u_{i+n}^L$. In the latter case, this path only visits vertices $u_r^\ell$ with $r > n$. Therefore, $n$ of the $2n$ paths are confined in this half of the graph, while the other $n$ paths are confined in the other half. More specifically, for each pair of complementary literals, one only concerns vertices $u_r^\ell$ with $r > n$ and the other only vertices with $r \leq n$. The clause vertices can only be visited by paths corresponding to satisfying literals. Thus, a feasible solution to the *MMBA* identifies a satisfying truth assignment.

## 3  A model

The problem admits the following mathematical formulation. Let $y_{ij}^\ell = 1$ if a worker of any group is assigned to shift $i$ in day $\ell$ and to shift $j$ in day $\ell + 1$, 0 otherwise. Let $x_{ij}^{k\ell} = 1$ if a worker of group $k$ is assigned to shift $i$ in day $\ell$ and to shift $j$ in day $\ell + 1$, 0 otherwise. Note that $x_{ij}^{k\ell}$ is undefined when the workers of class $k$ are unable to perform either shift $i$ in day $\ell$ or shift $j$ in day $\ell + 1$. Let $s_i^\ell$ be the total workload from day 1 to day $\ell$ of the worker performing shift $i$ in day $\ell$, and $z$ the maximum total workload over all workers.

$$\mathbf{P_1} : \min \quad z$$

$$\text{s.t.} \quad \sum_{j \in S_i^\ell} y_{ij}^\ell = 1 \qquad\qquad i = 1, \ldots, n,\ \ell = 1, \ldots, L - 1 \qquad (1)$$

$$\sum_{i \in P_j^\ell} y_{ij}^\ell = 1 \qquad\qquad i = 1, \ldots, n,\ \ell = 1, \ldots, L - 1 \qquad (2)$$

$$\sum_{k=1}^{K} x_{ij}^{k\ell} = y_{ij}^\ell \qquad\qquad \left(u_i^\ell, u_j^{\ell+1}\right) \in E \qquad (3)$$

$$\sum_{j \in P_i^{\ell-1}} x_{ji}^{k\ell} = \sum_{j \in S_i^\ell} x_{ij}^{k\ell} \qquad u_i^\ell \in V_2, \ldots, V_{L-1},\ k = 1, \ldots, K \qquad (4)$$

3

$$\sum_{i=1}^{n} \sum_{j \in S_i^1} x_{ij}^{k1} = n_k \qquad\qquad k = 1, \ldots, K \qquad\qquad (5)$$

$$s_i^1 = w_i^1 \qquad\qquad\qquad i = 1, \ldots, n \qquad\qquad\qquad (6)$$

$$s_i^\ell \geq w_i^\ell + \sum_{j \in P_i^{\ell-1}} s_j^{\ell-1} y_{ji}^{\ell-1} \quad i = 1, \ldots, n,\ \ell = 2, \ldots, L \qquad (7)$$

$$z \geq s_i^L \qquad\qquad\qquad i = 1, \ldots, n \qquad\qquad\qquad (8)$$

$$x_{ij}^{k\ell} \in \{0, 1\} \qquad\qquad \left(u_i^\ell, u_j^{\ell+1}\right) \in E,\ k = 1, \ldots, K \qquad (9)$$

$$y_{ij}^\ell \in \{0, 1\} \qquad\qquad \left(u_i^\ell, u_j^{\ell+1}\right) \in E \qquad\qquad\qquad (10)$$

### 3.1 A Lagrangean approach

If one relaxes in a Lagrangean fashion constraints (3), the problem decomposes in two classes of subproblems. The former includes $K$ min-cost flow problems, which only concern the $x_{ij}^{k\ell}$ variables

$$\mathbf{L_1^k} : \min \qquad\qquad \sum_{(u_i^\ell, u_j^{\ell+1}) \in E} \lambda_{ij}^\ell x_{ij}^{k\ell} \qquad\qquad (11)$$

$$\text{s.t.} \qquad\qquad (4)\ (5)\ (9)$$

where the integrality constraints (9) are redundant. Also notice that the underlying graphs are acyclic.

The latter is an unusual assignment problem, concerning the $y_{ij}^\ell$, $s_i^\ell$ and $z$ variables:

$$\mathbf{L_2} : \min \qquad\qquad z + \sum_{(u_i^\ell, u_j^{\ell+1}) \in E} \lambda_{ij}^\ell y_{ij}^\ell \qquad\qquad (12)$$

$$\text{s.t.} \qquad\qquad (1)\ (2)\ (6)\ (7)\ (8)\ (10)$$

whose objective function is composed of two terms. The first term leads to the bottleneck assignment problem introduced in Carraresi and Gallo (1984), for which an asymptotically optimal approximation algorithm exists. The second one leads to a classical assignment problem: constraints (6), (7) and (8) are redundant since $z$ is here a free variable.

To solve $\mathbf{L_2}$, we obtain a lower bound by optimizing separately the bottleneck and the min-cost subproblems, and summing the resulting optimal values. Notice that the bottleneck subproblem, which is the harder of the two, does not depend on the Lagrangean multipliers $\lambda_{ij}^\ell$. Therefore, even if the multipliers are iteratively updated, e. g. by a subgradient procedure, it is solved once

for all at the beginning of the computation. On the contrary, the min-cost subproblem needs to be solved at each step.

We observe that the algorithms for both the bottleneck and the min-cost assignment problems are based on the improvement of a given starting solution through the determination of augmenting paths. Adapting these procedures, it is possible to obtain an upper bound for the problem $\mathbf{L_2}$.

# 4    Conclusions

In this work, we have presented a new problem, derived from a real-world application, which extends the classical paper by Carraresi and Gallo (1984) on the Multilevel Bottleneck Assignment Problem. We have also outlined a possible solution scheme, based on a Lagrangean approach, which provides lower and upper bounds on the optimum.

Another possible solving approach could take into account the special "onion-like" structure in which $T_1 \subseteq \ldots \subseteq T_K$. This case is not furtherly discussed here, because it does not hold in our practical application, though it could characterize other real-world cases.

A different approach can consider a two-phase model: the first one distributes the shifts among the workers' classes; then, a second phase models, for each class, the assignment of the shifts to the workers. Following this approach, approximation algorithms can be derived.

# References

J. E. Beasley and B. Cao (1998), A Dynamic Programming based algorithm for the Crew Scheduling Problem, *Computers & Operations Research* 53, 567–582

P. Cappanera and G. Gallo (2003), On the Airline Crew Rostering Problem, *to appear in Operations Research*

A. Caprara, P. Toth, D. Vigo and M. Fischetti (1998), Modeling and Solving the Crew Rostering Problem, *Operations Research* 46, 820–830

P. Carraresi and G. Gallo (1984), A Multi-level Bottleneck Assignment Approach to the bus drivers' rostering problem, *European Journal of Operational Research* 16, 163–173

M. Guignard and S. Kim (1987), Lagrangean decomposition: a model yielding stronger Lagrangean bounds, *Mathematical Programming* 39, 215-228.