On Solving the Assignment Problem with Conflicts

Roberto Montemanni
Department of Sciences and Methods for Engineering
University of Modena and Reggio Emilia
Reggio Emilia, Italy
roberto.montemanni@unimore.it

Derek H. Smith
Faculty of Computing, Engineering and Science
University of South Wales
Pontypridd, Wales, UK
derek.smith@southwales.ac.uk

***Abstract*—A variant of the well-known Assignment Problem is studied in this paper, where pairs of assignments are conflicting, and cannot be selected at the same time. This configures a set of hard constraints. The problem, which models real applications, looks for a complete assignment that minimizes the total cost, while no conflict is violated.**

In this paper, we consider a previously known mixed integer linear program representing the problem and we solve it with the open-source solver CP-SAT, part of the Google OR-Tools computational suite.

An experimental campaign on the instances available from the literature, indicates that the approach we propose achieves results comparable with, those of state-of-the-art solvers, notwithstanding its intrinsic conceptual and implementation simplicity. The solver adopted is also able to provide heuristic solutions quicker and better than the heuristic methods previously discussed in the literature.

***Index Terms*—assignment problem, conflict constraints, exact solutions, heuristic solutions**

I. Introduction

The Assignment Problem is a well-known optimization problem with direct applications in several fields such as personnel scheduling, task assignment, job shop loading, facility location and workforce planning. Instances of any practical size can be solved efficiently in polynomial time using, for example the Hungarian algorithm [1], [2].

The focus of this work is on the Assignment Problem with Conflicts (APC), a problem introduced recently in [3], which is an extension of AP with additional conflict constraints that forbid pairs of assignments from happening at the same time, imposing therefore hard constraints. The APC deals with finding a minimum weight perfect matching such that no more than one edge is selected from each conflicting edge pair.

Classic optimization problems with conflicts have been studied extensively in the last decades, due to their practical implications, and their extended computational complexity in their general settings over the standard problems. For example, it is possible to trace studies on knapsack problems with conflicts [4], [5], on spanning trees with conflicts [6], [7], [8], [9], [10], on shortest paths with conflicts [11], set coverings with conflicts [12], [13] and maximum flows with conflicts [14], [15].

There are several practical applications of the APC. For example in container terminals, where arriving containers (V_A) have to be organized in the yard locations (V_B). Containers with certain characteristics – typically related either to size, weight or priorities – cannot be stacked on top of each others, leading therefore to conflict assignments. The costs of the assignment can be related to other real-world factors such as distance from the ship and the yard location. These settings configure an APC to be solved in order to optimize yard operations. Other similar applications of the APC are in general warehouses or multi-compartment vehicles, where goods with certain characteristics cannot be stored next to each other (e.g. food and toxic products). In personnel scheduling problems (e.g. airlines rostering) there might be incompatibilities between people, that should therefore not be assigned to the same team. This can be translated into straightforward conflicts within an APC.

Theoretical results on the computational complexity of the APC, together with some approximation results, can be found in [16]. Further results on special polynomially-solvable settings were presented in [17], together with some heuristics and lower bounding schema. More contribution for the APC can be found in [18] and [19], where some Mixed Integer Linear Programming models, exact Branch&Bound methods and heuristic ideas are introduced. The concepts of the last two papers were later summarized and extended in [3], which remains the reference work for the problem under investigation.

In this paper, a mixed integer linear programming model for the APC is considered and solved by the open-source solver CP-SAT, which is part of the Google OR-Tools [20] optimization suite. Successful application of this solver on optimization problems with characteristics similar to the problem under investigation, motivated our study [21], [22], [23]. An experimental campaign on the benchmark instances previously proposed in the recent literature is also presented and discussed.

The overall organization of the paper can be summarized as follows. The Assignment Problem with Conflicts is formally defined in Section II. Section III discusses a mixed integer linear programming model to represent the problem. In Section IV the approach we propose is compared with recent state-of-the-art methods from the literature. Conclusions are finally drawn in Section V.

II. Problem Description

The Assignment Problem with Conflicts (APC) can be formally described as follows. Let $G = (V_A \cup V_B, E)$ be a complete bipartite graph, where $|V_A| = |V_B|$, $V_A \cap V_B = \emptyset$, and $E = \{\{i,j\} | i \in V_A, j \in V_B\}$ is the set of possible assignments. A non-negative cost c_{ij} is given for each $\{i,j\} \in E$. Moreover, a set $C = \{\{i,j\},\{k,l\} | \{\{i,j\},\{k,l\}\} \in E\}$ represents the conflicts among pairs of assignments.

The objective of the APC is to find the perfect matching of minimum cost between the sets V_A and V_B that uses at most one edge from every conflict of C.

A small example of an APC instance and a respective feasible solution are depicted in Figure 1.

III. A Mixed Integer Linear Programming Model

In this section, a model for the APC, previously discussed in [3], is presented. A variable x_{ij} takes value 1 if the edge between $i \in V_A$ and $j \in V_B$ is selected, 0 otherwise. The resulting model is as follows.

$$
\begin{aligned}
\min \quad & \sum_{\{i,j\} \in E} c_{ij} x_{ij} && && (1)\\
s.t. \quad & \sum_{j:\{i,j\} \in E} x_{ij} = 1 && i \in V_A && (2)\\
& \sum_{i:\{i,j\} \in E} x_{ij} = 1 && j \in V_B && (3)\\
& x_{ij} + x_{kl} \le 1 && \{\{i,j\},\{k,l\}\} \in E && (4)\\
& x_{ij} \in \{0,1\} && \{i,j\} \in E && (5)
\end{aligned}
$$

The objective function (1) minimizes the cost of the assignment selected. Constraints (2) impose that each element of V_A has to be assigned to exactly one element of V_B through the feasible edges. Constraints (3) impose that each element of V_B has to be assigned to exactly one element of V_A through the feasible edges. Inequalities (4) model the conflicts, imposing that at most one of two conflicting edges can be selected. The domains of the x variables are finally specified in constraints (5).

IV. Computational Experiments

In Section IV-A we describe the benchmark instances previously introduced in the literature, and used for the present study. In Section IV-B the approach we propose is compared with the other methods available in the literature.

A. Benchmark Instances

In the literature, the only available benchmark set for the APC is – to our knowledge – the one proposed in [3]. We will therefore adopt these instances for our experiments. The number of nodes in the left hand-side of the bipartite graph $G = (V1(G) \cup V2(G), E(G))$, indicated as $|V(G)|$, takes values between 15 and 500. Conflict pairs are generated at random, and the number $|E(C)|$ of such pairs is between 5000 and 200000. A total of 135 test problems were introduced, but only 130 can be used for the experiments reported in this work, due to some inconsistencies in the available dataset.

We refer the interested reader to [3] for a comprehensive description of the instances.

B. Experimental Results

The model discussed in Section III has been solved with the Google OR-Tools CP-SAT solver [20] version 9.12. The experiments have been run on a computer equipped with an Intel Core i7 12700F CPU. The experiments for the methods previously appeared in [5] and against which we compare, were run on an a machine equipped with a 2.2 GHz Intel Core i7 processor (no more information is available), which according to http://gene.disi.unitn.it/test/cpu_list.php is 3 or more times slower.

The methods involved in the comparison are:

- LS: Local Search heuristic algorithm discussed in [3] (the original concepts of the method had already been discussed in [19]);
- RDS: Russian Doll Search heuristic algorithm discussed in [3];
- BIP: best results obtained by solving with CPLEX 12.7 [24] the two Binary Linear Programs presented in [3] (the model had already been introduced in [18] and [19]);
- B&B: Branch-and-Bound approach (with the best settings) presented in [3] (a preliminary version of the method appeared in [19]);
- CP-SAT: the mixed integer linear program presented in Section III solved with Google OR-Tools CP-SAT solver 9.12 [20].

The results are summarized in Table I, where for each group of five instances identified by the values of $|V_A|$ and $|C|$, the average of the optimal solutions is reported (*Opt*). All the methods considered were run for a maximum of 3600 seconds on each instance, and for each group of instances the following data are reported:

- Gap %: the average optimality, calculated as $100 \cdot \frac{Val-Opt}{Opt}$, where Val is the value returned by the method. This value is reported only for heuristic methods;
- Sec Best: the average computation time in seconds required to retrieve the best solution found. This value is reported only for heuristic methods;
- Sec Tot: the average computation time in seconds required to prove the optimality of the best solution found. This value is reported only for exact methods.

The last lines of the table contain the averages of the indicators tabled for the different methods.

The results suggest the following considerations. First, the Local Search heuristic is the only option when very quick solutions are required, although the quality of such solutions is typically a few percentage point off the optimum. In case longer computation times are allowed, CP-SAT is the best option, able to find all the optimal solutions in less time than the RDS heuristic, that also show a small but not null

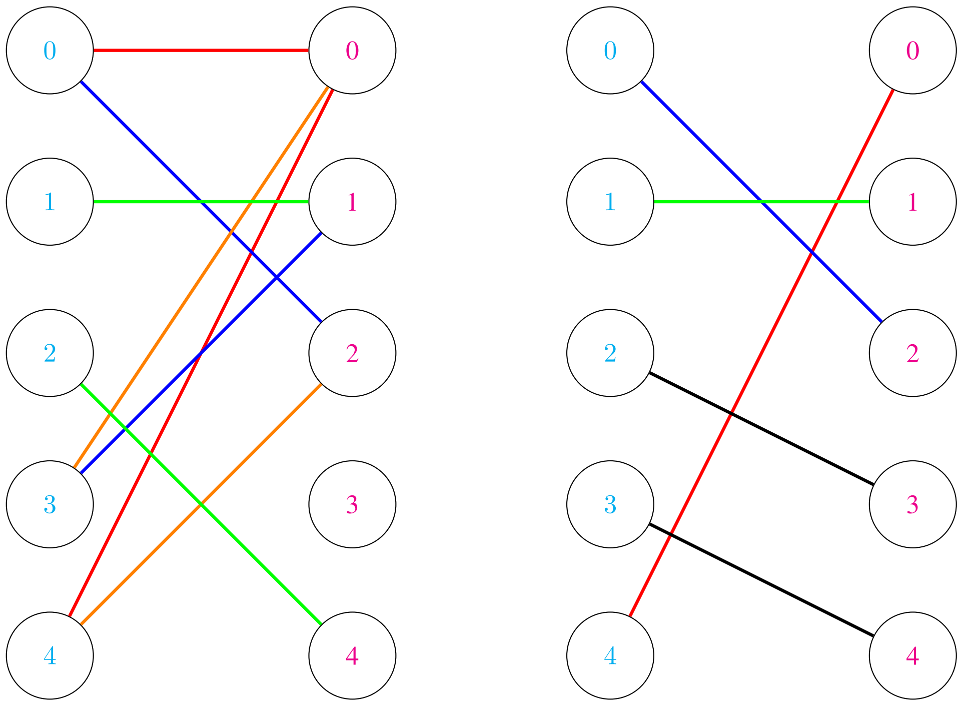

Fig. 1: On the left an example of a APC instance is presented, where conflicts are indicated as colored edges. Costs are omitted. On the right, a feasible solution is represented. Observe that at most one edge is used for each color (conflict) and some black edges (not involved in any conflicts) are added to complete the matching.

optimality gap. However – also considering the speed of the computers used for the experiments – the exact method BIP seems the best option, since it is able to provide proven optimal solutions quickly, although the method has the advantage of taking the best of two solutions. When comparing the three exact methods, they present comparable average computation times, but the behaviour on the different problems appears different: BIP is the more robust method, with balanced times among most of the instances, although the method suffers on the larger instances; B&B is very fast on the easier instances but it suffers heavily on the difficult instances; CP-SAT shows balanced performance but with over-long computation times on a few groups of instances that deteriorate the average performance.

In conclusion, BIP seems the best option for all the instances apart from the largest ones, on which B&B scales better and should be preferred. CP-SAT positions itself in between the two methods with an overall balanced behaviour.

V. Conclusions

A formulation based on Mixed Integer Linear Programming for the Assignment Problem with Conflicts has been considered and solved via the open-source solver CP-SAT, part of the Google OR-Tools computational suite.

The experimental results indicate that the approach we propose has performances comparable with those of the state-of-the-art exact solvers available in the literature, notwithstanding the minimal implementation effort required for our solution. Moreover, the CP-SAT solver used as a heuristic method appears to be superior to the proper heuristic methods previously introduced in the literature.

Acknowledgments

The work was partially supported by the Google Cloud Research Credits Program.

References

[1] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin, *Network flows*. Massachusetts Institute of Technology press, 1993.
[2] H. W. Kuhn, “The Hungarian method for the assignment problem,” *Naval Research Logistics Quarterly*, vol. 1-2, no. 2, pp. 83–97, 1955.
[3] T. Öncan, Z. Şuvak, M. H. Akyüz, and I. K. Altınel, “Assignment problem with conflicts,” *Computers & Operations Research*, vol. 111, pp. 214–229, 2019.
[4] A. Bettinelli, V. Cacchiani, and E. Malaguti, “A branch-and-bound algorithm for the knapsack problem with conflict graph,” *INFORMS Journal on Computing*, vol. 29, no. 3, pp. 457–473, 2017.
[5] S. Coniglio, F. Furini, and P. San Segundo, “A new combinatorial branch-and-bound algorithm for the knapsack problem with conflicts,” *European Journal of Operational Research*, vol. 289, no. 2, pp. 435–455, 2021.
[6] R. Zhang, S. Kabadi, and A. Punnen, “The minimum spanning tree problem with conflict constraints and its variations,” *Discrete Optimization*, vol. 2, no. 8, pp. 191–205, 2011.
[7] P. Samer and S. Urrutia, “A branch and cut algorithm for minimum spanning trees under conflict constraints,” *Optimization Letters*, vol. 1, no. 9, pp. 41–55, 2014.
[8] F. Carrabs, C. Cerrone, and R. Pentangelo, “A multi-ethnic genetic approach for the minimum conflict weighted spanning tree problem,” *Networks*, vol. 2, no. 74, pp. 134–147, 2019.
[9] F. Carrabs, R. Cerulli, R. Pentangelo, and A. Raiconi, “Minimum spanning tree with conflicting edge pairs: a branch-and-cut approach,” *Annals of Operations Research*, no. 298, pp. 65–78, 2019.

TABLE I: Computational results.

Instances		Opt	LS [3]		RDS [3]		BIP [3]	B&B [3]	CP-SAT	
$\lvert V_A \rvert$	$\lvert C \rvert$		Gap %	Sec Best	Gap %	Sec Best	Sec Opt	Sec Opt	Sec Best	Sec Opt
15	5000	2246.2	1.81	0.7	0.00	3.6	13.2	1.4	0.2	1.7
20	10000	2450.8	1.99	0.8	0.00	13.1	130.1	6.3	17.5	886.7
30	20000	3247.0	2.13	0.9	0.00	11.5	25.8	3.0	21.5	117.5
30	30000	3355.8	2.18	1.1	0.00	637.4	1650.3	129.0	623.2	1931.1
40	40000	4205.0	2.17	1.8	0.00	24.7	12.2	3.5	17.9	56.5
50	50000	5183.4	2.10	2.5	0.00	39.4	7.6	3.3	24.4	38.8
50	60000	5186.2	2.22	2.9	0.00	197.5	14.9	3.9	36.8	68.6
60	80000	6163.6	2.25	3.7	0.00	77.6	10.6	5.6	42.2	60.2
70	100000	7156.8	2.31	3.9	0.00	54.7	9.2	2.6	29.6	38.4
70	150000	7166.8	2.44	4.1	0.01	2070.7	46.7	104.4	118.2	204.8
80	200000	8154.2	2.43	4.3	0.05	2667.9	34.4	141.0	68.8	104.9
90	250000	9157.2	2.45	4.3	0.07	3599.6	68.5	1126.9	223.1	359.5
100	100000	10118.8	2.47	4.2	0.00	6.5	7.8	0.9	11.4	12.3
100	250000	10144.0	2.49	4.5	0.01	1242.8	23.9	875.5	72.1	86.6
100	350000	10160.0	2.61	4.7	0.42	3599.4	149.9	3171.5	451.0	832.6
150	200000	15093.6	2.87	5	0.00	1.3	15.3	0.4	22.0	22.5
150	350000	15102.2	2.93	5.1	0.00	117.5	22.7	7.1	32.9	34.1
150	500000	15105.8	3.16	6.5	0.00	337.1	33.8	38.1	54.5	55.9
200	200000	20077.4	3.54	8.2	0.00	0.4	39.3	0.1	25.5	26.5
200	400000	20082.8	3.68	8.6	0.00	6.5	46.5	1.7	38.5	39.6
250	500000	25058.2	4.08	10.5	0.00	6.6	99.5	0.9	50.2	51.4
250	700000	25057.6	4.33	11.3	0.00	2.9	106.8	2.5	62.4	64.2
300	100000	30042.4	4.49	12.8	0.00	0.2	184.1	0.1	31.9	32.9
300	300000	30040.6	4.72	16	0.00	0.6	188.6	0.1	40.7	41.9
400	200000	40016.6	5.16	20.5	0.00	0.3	608.8	0.0	57.9	59.7
500	200000	50006.2	7.12	23.7	0.00	6.9	1497.3	0.0	85.7	87.7
Averages			3.08	6.6	0.02	566.4	194.1	216.5	141.6	204.5

[10] F. Carrabs and M. Gaudioso, "A lagrangian approach for the minimum spanning tree problem with conflicting edge pairs," *Networks*, vol. 1, no. 78, pp. 32–45, 2021.

[11] R. Cerulli, G. Guerriero, E. Scalzo, and C. Sorgente, "Shortest paths with exclusive-disjunction arc pairs conflicts," *Computers & Operations Research*, vol. 152, p. 106158, 2023.

[12] S. Saffari and Y. Fathi, "Set covering problem with conflict constraints," *Computers & Operations Research*, vol. 143, p. 105763, 2022.

[13] F. Carrabs, R. Cerulli, R. Mansini, L. Moreschini, and D. Serra, "Solving the set covering problem with conflicts on sets: A new parallel GRASP," *Computers & Operations Research*, vol. 166, p. 106620, 2024.

[14] Z. Şuvak, I. K. Altınel, and N. Aras, "Exact solution algorithms for the maximum flow problem with additional conflict constraints," *European Journal of Operational Research*, vol. 287, no. 2, pp. 410–437, 2020.

[15] F. Carrabs, R. Cerulli, R. Mansini, D. Serra, and C. Sorgente, "Hybridizing carousel greedy and kernel search: A new approach for the maximum flow problem with conflict constraints," *European Journal of Operational Research*, 2025.

[16] A. Darmann, U. Pferschy, and G. Schauer, J.and Woeginger, "Paths, trees and matchings under disjunctive constraints," *Discrete Applied Mathematics*, vol. 16, no. 159, pp. 1726–1735, 2011.

[17] T. Öncan, R. Zhang, and A. P. Punnen, "The minimum cost perfect matching problem with conflict pair constraints," *Computers & Operations Research*, vol. 40, no. 4, pp. 920–930, 2013.

[18] T. Öncan and I. K. Altınel, "Iterated exact and heuristic algorithms for the minimum cost bipartite perfect matching problem with conflict constraints," in *2017 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM)*, 2017, pp. 1032–1036.

[19] ——, "A branch-and-bound algorithm for the minimum cost bipartite perfect matching problem with conflict pair constraints," *Electronic Notes in Discrete Mathematics*, vol. 64, pp. 5–14, 2018, 8th International Network Optimization Conference - INOC 2017.

[20] L. Perron and F. Didier, "Google OR-Tools - CP-SAT," Google, 2025, https://developers.google.com/optimization/cp/cp_solver/.

[21] R. Montemanni and M. Dell'Amico, "Solving the parallel drone scheduling traveling salesman problem via constraint programming," *Algorithms*, vol. 16, no. 1, p. 40, 2023.

[22] R. Montemanni, M. Dell'Amico, and A. Corsini, "Parallel drone scheduling vehicle routing problems with collective drones," *Computers & Operations Research*, vol. 163, p. 106514, 2024.

[23] R. Montemanni, "Solving a home healthcare routing and scheduling problem with real-world features," in *Proceedings of the 9th International Conference on Machine Learning and Soft Computing*. Springer, to appear, 2025.

[24] IBM, "IBM CPLEX Optimizer," 2024, https://www.ibm.com/de-de/analytics/cplex-optimizer [Accessed: 2024-03-14].