Ant-based algorithms for minimal perturbation problem of hotel scheduling

Seminar: Recent Results in Swarm Intelligence

Milan Rybář
kontakt@milanrybar.cz

1 Introduction

Scheduling is a well-known optimization problem containing many diverse domains, for instance timetable scheduling problem, job shop problem, and nurse scheduling problem. Most of them are at least NP-hard. In this paper, we focus on creating a schedule for a hotel. Hotels can define some constrains that the schedule must satisfy. One of possible constrains is called minimal perturbation problem for which a new booking should change the current schedule as little as possible.

2 Minimal perturbation problem of hotel scheduling

Let's assume that a hotel contains R identical rooms and we have a valid schedule $S^O \in \{1, \dots, R\}^B$ of B bookings for these rooms, where S_i^O represents a room's number for the booking i. Every booking is defined by its start and length. We are given a new booking and our goal is to create a new valid schedule $S^N \in \{1, \dots, R\}^{B+1}$, where each booking is assigned to a room and they do not overlap, and which has a minimal number of perturbations of bookings in comparison with the original schedule. Additionally, we assume that such a valid schedule exists. Figure 1 shows a visual definition and a solution for one problem's instance.

Formally, this is a optimization problem defined as

$$\begin{aligned}
&\min_{S^N} && \sum_{i=1}^{B} \|S_i^N \neq S_i^O\| \\
&\text{subject to} && \sum_{i,j \in \{1,\dots,B+1\}, i \neq j} \text{overlap}(S^N, i, j) = 0 \\
&\text{where} && \text{overlap}(S, i, j) = \begin{cases} 1 & \text{if bookings } i \text{ and } j \text{ overlap in schedule } S \\ 0 & \text{otherwise} \end{cases} \\
& && \|\text{expression}\| = \begin{cases} 1 & \text{if expression is true} \\ 0 & \text{otherwise} \end{cases}
\end{aligned}$$

2.1 Graph representation

This problem can be re-defined as a graph coloring problem with R colors where vertices represents bookings and edge between two bookings exists if they would overlap in the same room. In this representation, a color defines a room for the booking and a (valid) graph's coloring defines a (valid) schedule.

Rooms/Days	1	2	3	4	5	6	7	8	9	10
1	-	1	1	1	-	2	2	2	-	3
2	4	4	-	5	5	-	6	6	6	6
3	-	7	7	7	7	7	8	8	8	8
4	9	9	9	-	10	10	10	-	11	11
New Booking	-	-	12	12	12	12	-	-	-	-

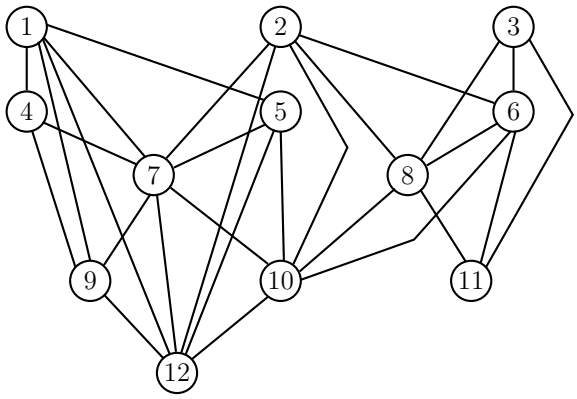
New Schedule:
Number of Perturbations: 3
Room for New Booking: 2

Rooms/Days	1	2	3	4	5	6	7	8	9	10
1	9	9	9	5	5	2	2	2	-	3
2	4	4	12	12	12	12	6	6	6	6
3	-	7	7	7	7	7	8	8	8	8
4	-	1	1	1	10	10	10	-	11	11

Fig. 1: Visual definition and solution for *simple* problem's instance.

Formally, we have an undirected graph $G = (V, E)$ with $V = \{1, \ldots, B + 1\}$ and $(i, j) \in E \Leftrightarrow \text{overlap}(S^1, i, j) = 1$, where $\forall v \in V : S^1_v = 1$, and a graph's coloring $e \colon V \to \{1, \ldots, R\}$.

Graph coloring problem primary focuses on finding a chromatic number representing the minimal number of colors to color the graph. Nevertheless, we know that a graph's coloring with R colors exists and our objective is to found the one among them that minimize the number of perturbations of bookings in comparison with the original schedule. We start with the graph's coloring defined by the initial schedule S_O, $\forall v \in \{1, \ldots, B\} \colon e(v) = S^O_v$. Only the new booking has no color, $e(B + 1)$ is undefined, and we must re-color the graph while keeping a minimal number of perturbations of bookings. The figure 2 shows a graph representation for one problem's instance defined at the figure 1.



$e(1) = e(2) = e(3) = 1$
$e(4) = e(5) = e(6) = 2$
$e(7) = e(8) = 3$
$e(9) = e(10) = e(11) = 4$
$e(12)$ is undefined

Fig. 2: Graph representation for *simple* problem's instance defined at the figure 1. The initial graph's coloring e is defined by the initial schedule S_O. The new booking (vertex 12) has no color yet.

It can be also seen as a multi-optimization problem of minimizing a number of conflicts (#C) and a number of perturbations (#P).

$$\min_{e}\Big(\underbrace{\sum_{v\in V}\Big\|\sum_{(v,n)\in E}\|e(v)=e(n)\|>0\Big\|}_{\text{number of conflicts (\#C)}},\underbrace{\sum_{v\in V\setminus\{B+1\}}\|e(v)\neq S_v^O\|}_{\text{number of perturbations (\#P)}}\Big)$$

Nevertheless, we keep in mind our original problem of hotel scheduling, for which we need an algorithm that minimizes #C to 0.

3 State of the art of ant-based algorithms

3.1 Graph coloring problem

Graph coloring problem primary focuses on finding a chromatic number. Therefore, ant-based algorithms contain a number of allowed colors to use. If they find a feasible coloring, they decrease this number and try to find a new coloring with less colors. They differs in using ants, selection of the next vertex and color, and whether they use pheromones or not.

We can divide them to two categories: online and offline algorithms. In online version, an ant immediately colors the graph during its motion. On the other hand in offline version, the graph is re-colored after all ants finished their movements.

Offline algorithms The first ant-based approach for a graph coloring problem was proposed by Costa and Hertz [5]. An ant is allowed to move from one vertex to any other vertex in the graph and colors the full graph in a constructive way. The pheromone trail for two vertices represents a weight to assign them the same color. Their approach does not perform well against traditional graph techniques [9].

Modification by Bessedik at al. [2] uses ant colony system instead of basis ant system.

Online algorithms An ant in the algorithm by Comellas and Ozon [4] moves along the edges of the graph in cycles and uses a local information to color a vertex. Another difference to Costa and Hertz's algorithm is that all ants work together on a single graph's coloring and they do not have pheromone laying capability.

On the contrary in the approach by Bui at al. [3], each ant colors only a portion of the graph. Again in this algorithm, the ants do not use pheromones.

3.2 Constraint programming

Ant-based algorithms for constrain programming are mainly used for over-constrained domains where the main problem is to find a feasible solution satisfying hard constrains and soft constrains are not so important in this context [8] [7]. Thus, these algorithms are not suitable for our case.

3.3 Scheduling

Even thought our topic is a subset of a timetable problem, we have less constrains that a typical timetable problem solved by ant-based algorithms [6] [1].

4 Our approach

In this paper, we try both approaches for a graph coloring problem presented in 3.1, namely one online and two offline ant-based algorithms. Unlike ants in the offline version, that use pheromones to exchange information between them, ants in the online version do not have pheromone laying capability, nevertheless, immediate change of graph's coloring can be considered as a form of stigmergy.

Firstly, we did preliminary testing on a few problem's instances. Results of these experiments are not included. From that we conclude possible set of good parameters we use for the testing the rest of problem's instances.

All of our ant-based algorithms share a few similar following properties:

Fitness function In order to get the best solution, we must define a fitness function for an ant to describe a quality of its solution. We keep in mind our original problem of hotel scheduling, hence we want firstly create a schedule with almost no conflicts, and then we can focus on minimizing the number of perturbations. Therefore, we define the ant's fitness value as $\#C * N + \#P$ where $N = B + 1$ is number of vertices of the graph. We also tried $\#C + \#P$ and $\#C + \frac{\#P}{\#C+1}$, but they did not perform well.

Movement Ants can only move on a portion of the graph limited by a number of steps they can make. This makes sense in our case, because we do not want to re-color the whole graph so easily since it could cause a huge number of perturbations of bookings. Also, each ant has a memory of visited vertices and it chooses the next vertex to move to only from its unvisited vertices.

Coloring When an ant chooses a color for a particular vertex, it does not consider the color from the previous colored vertex. This was used in Bui at al.'s approach and it prevents additional conflicts to previous vertices.

Initial vertex An ant is put to the last vertex representing the new booking if it still has no color, otherwise it randomly chooses a vertex from all conflicting vertices.

5 Online ant-based algorithm

Inspired by Bui at al.'s algorithm, an ant moves on the graph presented before and colors only a portion of the graph by a local heuristic. Each ant makes two decisions, firstly it chooses a vertex to move to and then a color for this vertex. The pseudocode for this algorithm is shown at the algorithm 1.

5.1 Choice of vertex

The ant chooses the next vertex only among adjacent vertices with any conflict. If there is not such a vertex, the ant does not move.

Policy	Description	Performance
Bui at al.	Each ant at Bui at al.'s algorithm moves by a path of length two. Firstly, the ant chooses a random vertex and then a vertex with the most conflicts.	poor
RV	Random vertex	ok
MC	Vertex with the most conflicts	ok
Rank-based	Each ant randomly chooses a vertex according to its probability defined as a number of vertex's conflicts ($\#C$).	poor

Algorithm 1: Pseudocode for online ant-based algorithm

```
forall the iterations do
    if iterations without improvement > stagnation limit then
        Apply the policy for stagnation;
    forall the ants do
        forall the steps do
            Choose and move to the next vertex;
            Choose and color the current vertex;
    Update best and global best solution;
```

5.2 Choice of color

Policy	Description	Performance
Best	Each ant chooses the color that minimize the expression $\#C + PP$, where PP is a penalty for perturbation. It performs best for $PP = 0.5$.	ok
Rank-based	Each ant randomly chooses a color according to its probability defined as $\frac{1}{\#C+PP+1}$.	poor

5.3 Stagnation policy

Policy	Description	Performance
Reset	It resets the algorithm.	poor
Perturb	Bui at al.'s algorithm chooses top 10% vertices with most conflicts and randomly re-colors their neighbours.	ok

5.4 Parameters

We use the same parameters as at the Bui at al.'s paper.

- Number of ants: $\lceil 0.2 * N \rceil$
- Maximum number of steps: $\lceil \frac{N}{4} \rceil$
- Stagnation limit: 20
- Maximum number of iterations: 500

6 Ant colony system

In this offline ant-based algorithm, the ants use pheromones as in ant colony system (ACS). The ant on the previously presented graph must still make two decisions. Firstly, the ant chooses a vertex to move to and then a color for this vertex according to ACS rule containing pheromones. That means our graph contains pheromones for colors in every vertex. The pseudocode for this algorithm is shown at the algorithm 2. Each ant starts with an initial coloring defined as the initial graph's coloring at the current iteration.

Algorithm 2: Pseudocode for ACS-based algorithm

```
forall the iterations do
    if iterations without improvement > stagnation limit then
        Apply the policy for stagnation;
    forall the steps do
        forall the ants do
            Choose and move to the next vertex;
            Choose and color the current vertex by ACS rule;
            Online pheromone update;
    Update best and global best solution;
    Update graph's coloring according to the best solution;
    Offline pheromone update;
```

6.1 Choice of vertex

We tested the same policies as for online ant-based algorithm (5.1), but their performance differs.

Policy	Performance
Bui at al.	poor
RV	poor
MC	poor
Rank-based	ok

6.2 Choice of color

The ant chooses a color according to ACS rule. With a probability of q, it chooses a color with the highest probability, otherwise randomly chooses a color according to its probability, defined by a pheromone and heuristic value for this color. We consider following heuristic functions

Heuristic function	Definition	Description
HPR	$\frac{1}{\#C+1} + PR$	PR is a preference for the original color, which means being the original color. An ant minimize a number of conflicts and maximize a number of original colors.
HPP	$\frac{1}{\#C+PP+1}$	PP is a penalty for a perturbation, which means not being the original color. An ant directly minimize a number of conflicts plus a number of perturbations.
Dynamic	HPP with $PP = \frac{1}{\#C_{Best}+1}$	$\#C_{Best}$ is a number of conflicts of the best ant from previous iteration, otherwise 0. When there are a lot of conflicts, we set the penalty low, but when we are getting close to the valid schedule, we increase the penalty. This way, we include global information from the previous iteration into the ant's decision.
Scaled-dynamic	HPP with $PP = \frac{0.5}{\#C_{Best}+1}$	Experiments showed that algorithm performs best for $PP = 0.5$, therefore we scale the penalty to this interval.

6.3 Offline pheromone update

Ants move only by a limited number of steps, thus we also consider to update pheromone value for all ants instead of only the best ant. We test both of these variants.

6.4 Stagnation

Policy	Performance
Reset	ok
Perturb	poor
(It breaks the meaning of pheromones values.)	

6.5 Parameters

We use the same parameters as in the online ant-based algorithm (5.4). By experiments, we set ACS specific values:

- Initial pheromone value: 1
- Evaporation rate (φ): 0.7
- Heuristic preference coefficient (β): 2
- Preference for exploitation (q): 0.7

7 Ant colony system on the full graph

In the previous ACS-based algorithm, the ant have to make two decisions, to choose a vertex and a color. We extend the previously presented graph in order to get rid of this, so the ant has to make only one decision. The pseudocode for this algorithm is shown at the algorithm 3. In this graph, every color from the previous graph is a vertex and we define edges between these new vertices.

Formally, we have a graph $G_E(V_E, E_E)$ with $V_E = \{(b, c)\}$ for $\forall b \in \{1, \ldots, B+1\} \cup \{S\}, c \in \{1, \ldots, R\}$ and $((b_1, c_1), (b_2, c_2)) \in E_E \Leftrightarrow ((\text{overlap}(S^1, b_1, b_2) = 1 \wedge c_1 \neq c_2) \vee (b_1 = S \wedge b_2 = B+1) \vee (b_1 = B+1 \wedge b_2 = S))$. S represents a new starting vertex for the ants. Also we should notice that we do not create edges between the same colors.

When the ant moves to the vertex (v, c), it represents to color the vertex v by the color c. We remind that the ant chooses the next vertex only from unvisited adjacent vertices in the original graph, which means that after this move, the vertex v is marked as visited and the ant cannot visit another (v, i), $\forall i \in \{1, \ldots, R\}$.

7.1 Choice of vertex

We use almost the same heuristic functions as in the previous ACS-based algorithm, only we divide PP value by number of all colors, because the graph is bigger and otherwise the penalty by the original PP would be too large.

7.2 Offline pheromone update

As in the previous ACS-based algorithm, we test to update pheromones for all ants and for only the best ant as well.

Algorithm 3: Pseudocode for ACS-based algorithm on the full graph

```
forall the iterations do
    if iterations without improvement > stagnation limit then
        Apply the policy for stagnation;
    forall the steps do
        forall the ants do
            Choose and move to the next vertex by ACS rule;
            Online pheromone update;
    Update best and global best solution;
    Update graph's coloring according to the best solution;
    Offline pheromone update;
```

7.3 Parameters

We set the same parameters as in the previous ACS algorithm (6.5) except the following value:

- Preference for exploitation (q): 0.6

8 Experimental results

We run each algorithm on seven problem's instances shown at the table 1 in 50 trials. Summary statistics of these results are shown at the table 2 in appendix A. We analyze these results in this section.

Instance	#rooms	#days	#bookings in S_O	Length of the new booking	Minimum #P for #C=0
simple	4	10	11	4	3
simple_past	2	8	6	2	2
random_5x50	5	50	96	4	1
random_10x50	10	50	163	5	3
random_20x50	20	50	281	7	4
random_10x100	10	100	316	19	7
random_10x200	10	200	619	31	12

Table 1: Summary of problem's instances used for testing. (# = number of)

Problem's instances *simple* and *simple_past* were manually created and test the main functionality. Instance *simple* is shown at the figure 1 and *simple_past* proves that bookings in the past must also be rescheduled. The rest of five harder problem's instances were randomly created. Each room in these schedules is booked about 80% of its capacity.

The first thing we should notice for each algorithm is that a number of perturbations of the best solution improved significantly in comparison with the first solution with no conflicts. It justify of using our fitness function for ants.

8.1 Online ant-based algorithm

By changing PP value, we can see what an ant focus on. With $PP = 0$, the ant does not care about perturbations, therefore its goal is to found a solution without any conflicts. This results in a huge number of perturbations. By increasing PP, the ant takes also the number of perturbations into consideration. Nevertheless, this also increases probability to rise the number of conflicts. Value of $PP = 0.5$ produces best results, because values of 1 and 1.5 put too much pressure on a number of perturbation and the ant is not able to find a solution without any conflicts. For the future work, we suggest using adaptive change of PP, to start with 0 and then to increase it as the number of conflicts decreases.

Concluding from the most of results, random vertex (RV) policy for choosing the next vertex outperforms the policy of most conflicts (MC).

Despite its simplicity, the algorithm can find a reasonable suboptimal solution. Also, compared to ACS-based algorithms, it requires less computational time and less memory.

8.2 Ant colony system

In *simple*, *simple_past*, and *random_5x50* instances, each configuration of ACS found the optimum solution each time. Even on the rest of problem's instances, it beats online ant-based algorithm and ACS on the full graph. The best settings of heuristic function is HPR, the second one is HPP, and then dynamic and scaled-dynamic of HPP which do not differ too much from each another.

The decision, whether ACS should update pheromones for only the best ant or for all ants, is not so clear. We think it depends on the size of the graph. When the graph is small, updating pheromones is enough for the best ant, but when the size of the graph increases, it is better to update pheromones according to all ants, otherwise pheromones evaporate too quickly. For the future works, we suggest updating pheromones as in rank-based ant system and using adaptive change of PP and PR value, now it is fixed to value of 0.5.

As a side note, it is worth mentioning that it took about 2 hours to find the optimum solution for *random_10x200* instance in *SISCtus Prolog* by traditional constraint satisfaction approach. Nevertheless, ACS was able to find the optimum solution in about 20% of all runs.

8.3 Ant colony system on the full graph

ACS on the full graph performed the worst from our algorithms. We think it is due to the larger graph which needs more careful settings of parameters. For the future work, we also think that a candidate list would help, because the ant has to choose between a large number of options.

By focusing on minimizing the number of conflicts, the setting with HPP seems to be the best one. In most cases, it at least found a solution without any conflicts and the number of perturbations is about 2-3 times larger that in the previous ASC algorithm.

As conclusion, ACS on the extended graph with our settings is unsuitable for our problem. Furthermore, it needs more memory to store the extended graph with pheromones and it also requires more computational time due to the larger number of pheromones. On the other hand, the ant makes only one decision and the graph does not directly contain edges between the same colors, which previous algorithms must take care of explicitly.

9 Conclusion

We tested three different ant-based algorithms for minimal perturbation problem of hotel scheduling, namely online ant-based algorithm, ACS-based algorithm, and ACS-based algorithm on the full graph.

ACS-based algorithm found the optimum solution in more that 20% trials, even on the hardest problem's instance where two other algorithms never found the optimal solution. Therefore it is usable for the real application. By polishing its setting and implementing our suggestions, we believe it would improve even more. Furthermore, it can quickly find a suboptimal solution with no conflicts and then it can improve this solution if there is more time.

References

1. Abounacer, R., Boukachour, J., Dkhissi, B., Alaoui, A.E.H.: A hybrid ant colony algorithm for the exam timetabling problem. Revue ARIMA 12, 15–42 (2010)
2. Bessedik, M., Laib, R., Drias, A.: Ant colony system for graph coloring problem. In: Computational Intelligence for Modelling, Control and Automation, 2005 and International Conference on Intelligent Agents, Web Technologies and Internet Commerce, International Conference on. vol. 1, pp. 786–791. IEEE (2005)
3. Bui, T.N., Nguyen, T.H., Patel, C.M., Phan, K.A.T.: An ant-based algorithm for coloring graphs. Discrete Applied Mathematics 156(2), 190–200 (2008)
4. Comellas, F., Ozon, J., Comellas, F.: An ant algorithm for the graph colouring problem. In: in proceedings of the First International Workshop on Ant Colony Optimization. Citeseer (1998)
5. Costa, D., Hertz, A.: Ants can colour graphs. Journal of the operational Research Society 48(3), 295–305 (1997)
6. Eley, M.: Ant algorithms for the exam timetabling problem. In: Practice and Theory of Automated Timetabling VI, pp. 364–382. Springer (2007)
7. Meyer, B., Ernst, A.: Integrating aco and constraint propagation. In: Ant Colony Optimization and Swarm Intelligence, pp. 166–177. Springer (2004)
8. Solnon, C.: Ants can solve constraint satisfaction problems. Evolutionary Computation, IEEE Transactions on 6(4), 347–357 (2002)
9. Žerovnik, J., Vesel, A.: How well can ants color graphs? CIT. Journal of computing and information technology 8(2), 131–136 (2000)

A Experimental results

Experimental results of best solutions summarized over 50 trials are shown at the table 2. Prefix 'L -' represents values from the last iterations (500) and prefix 'F -' represents values from the first iterations in which appear a solution with no conflicts, otherwise the last iteration. Namely, 'L - Pert.' is a number of perturbations, 'L - Confl.' is a number of conflicts, and 'L - Stag.' is a number of stagnations from the last iteration. 'Opt' is how many times an algorithm finds the optimum solution from 50 trials. 'F - Iterations' is a number of the first iteration in which appears the solution with no conflicts and 'F - Pert.' is a number of perturbations, 'F - Confl.' is a number of conflicts, and 'F - Stag.' is a number of stagnations from this iteration.

Table 2: Summary statistics of experimental results of best solution in 50 runs.

simple — optimum solution #P = 3									
Algorithm	Settings	L - Pert.	L - Confl.	L - Stag.	Opt	F - Iterations	F - Pert.	F - Confl.	F - Stag.
Online	RV PP=0	3.00 ± 0.00	0.04 ± 0.28	1.78 ± 3.76	49	48.36 ± 80.04	3.04 ± 0.28	0.04 ± 0.28	1.78 ± 3.76
Online	MC PP=0	3.70 ± 1.43	0.00 ± 0.00	0.90 ± 1.20	40	32.96 ± 23.74	4.42 ± 1.81	0.00 ± 0.00	0.90 ± 1.20
Online	RV PP=0.5	2.88 ± 0.59	0.12 ± 0.48	7.06 ± 6.52	47	156.76 ± 137.55	2.96 ± 0.73	0.12 ± 0.48	7.06 ± 6.52
Online	MC PP=0.5	3.12 ± 0.59	0.00 ± 0.00	1.34 ± 1.67	48	40.64 ± 35.81	3.56 ± 1.28	0.00 ± 0.00	1.34 ± 1.67
Online	RV PP=1	2.40 ± 1.21	0.40 ± 0.81	11.62 ± 8.62	40	253.24 ± 183.57	2.44 ± 1.26	0.40 ± 0.81	11.62 ± 8.62
Online	MC PP=1	3.00 ± 0.00	0.00 ± 0.00	2.02 ± 2.06	50	55.20 ± 43.42	3.32 ± 0.74	0.00 ± 0.00	2.02 ± 2.06
Online	RV PP=1.5	1.74 ± 1.50	0.84 ± 1.00	14.00 ± 8.69	29	303.96 ± 188.12	1.74 ± 1.50	0.84 ± 1.00	14.00 ± 8.69
Online	MC PP=1.5	2.94 ± 0.42	0.04 ± 0.28	5.54 ± 5.85	49	126.88 ± 121.16	3.10 ± 0.71	0.04 ± 0.28	5.54 ± 5.85
ACS	HPR best	3.00 ± 0.00	0.00 ± 0.00	3.10 ± 3.50	50	43.70 ± 45.87	3.08 ± 0.40	0.00 ± 0.00	3.10 ± 3.50
ACS	HPR all	3.00 ± 0.00	0.00 ± 0.00	2.52 ± 2.69	50	35.84 ± 35.66	3.00 ± 0.00	0.00 ± 0.00	2.52 ± 2.69
ACS	HPP best	3.00 ± 0.00	0.00 ± 0.00	1.88 ± 2.03	50	27.68 ± 27.59	3.00 ± 0.00	0.00 ± 0.00	1.88 ± 2.03
ACS	HPP all	3.00 ± 0.00	0.00 ± 0.00	2.20 ± 3.02	50	31.60 ± 39.79	3.00 ± 0.00	0.00 ± 0.00	2.20 ± 3.02
ACS	Dynamic best	3.00 ± 0.00	0.00 ± 0.00	1.46 ± 2.22	50	21.58 ± 29.97	3.00 ± 0.00	0.00 ± 0.00	1.46 ± 2.22
ACS	Dynamic all	3.00 ± 0.00	0.00 ± 0.00	1.58 ± 2.07	50	23.70 ± 27.42	3.00 ± 0.00	0.00 ± 0.00	1.58 ± 2.07
ACS	Scaled-dynamic best	3.00 ± 0.00	0.00 ± 0.00	1.50 ± 2.33	50	22.52 ± 31.79	3.04 ± 0.28	0.00 ± 0.00	1.50 ± 2.33
ACS	Scaled-dynamic all	3.00 ± 0.00	0.00 ± 0.00	2.40 ± 2.83	50	34.68 ± 37.94	3.00 ± 0.00	0.00 ± 0.00	2.40 ± 2.83
ACSFull	HPP best	1.14 ± 1.47	1.24 ± 0.98	32.06 ± 12.99	19	390.66 ± 158.27	1.14 ± 1.47	1.24 ± 0.98	32.06 ± 12.99
ACSFull	HPP all	1.32 ± 1.50	1.12 ± 1.00	31.92 ± 13.64	22	388.80 ± 165.86	1.32 ± 1.50	1.12 ± 1.00	31.92 ± 13.64
ACSFull	Dynamic best	0.30 ± 0.91	1.80 ± 0.61	39.36 ± 6.95	5	479.78 ± 84.65	0.30 ± 0.91	1.80 ± 0.61	39.36 ± 6.95
ACSFull	Dynamic all	0.12 ± 0.59	1.92 ± 0.40	39.88 ± 5.55	2	486.32 ± 67.74	0.12 ± 0.59	1.92 ± 0.40	39.88 ± 5.55
ACSFull	Scaled-dynamic best	0.36 ± 0.98	1.76 ± 0.66	38.04 ± 8.46	6	463.88 ± 103.09	0.36 ± 0.98	1.76 ± 0.66	38.04 ± 8.46
ACSFull	Scaled-dynamic all	0.30 ± 0.91	1.80 ± 0.61	39.78 ± 5.13	5	484.86 ± 62.71	0.30 ± 0.91	1.80 ± 0.61	39.78 ± 5.13
simple_past — optimum solution #P = 2									
Algorithm	Settings	L - Pert.	L - Confl.	L - Stag.	Opt	F - Iterations	F - Pert.	F - Confl.	F - Stag.
Online	RV PP=0	2.28 ± 0.70	0.00 ± 0.00	0.02 ± 0.14	43	5.96 ± 7.10	2.28 ± 0.70	0.00 ± 0.00	0.02 ± 0.14
Online	MC PP=0	2.36 ± 0.78	0.00 ± 0.00	0.00 ± 0.00	41	1.90 ± 1.31	2.36 ± 0.78	0.00 ± 0.00	0.00 ± 0.00
Online	RV PP=0.5	2.36 ± 0.78	0.00 ± 0.00	0.10 ± 0.30	41	10.28 ± 11.02	2.36 ± 0.78	0.00 ± 0.00	0.10 ± 0.30
Online	MC PP=0.5	2.48 ± 0.86	0.00 ± 0.00	0.00 ± 0.00	38	2.36 ± 1.56	2.48 ± 0.86	0.00 ± 0.00	0.00 ± 0.00
Online	RV PP=1	2.16 ± 0.55	0.00 ± 0.00	0.02 ± 0.14	46	5.66 ± 6.09	2.16 ± 0.55	0.00 ± 0.00	0.02 ± 0.14
Online	MC PP=1	2.60 ± 0.93	0.00 ± 0.00	0.00 ± 0.00	35	2.08 ± 1.40	2.60 ± 0.93	0.00 ± 0.00	0.00 ± 0.00
Online	RV PP=1.5	2.28 ± 0.70	0.00 ± 0.00	0.06 ± 0.24	43	12.22 ± 10.35	2.28 ± 0.70	0.00 ± 0.00	0.06 ± 0.24
Online	MC PP=1.5	2.76 ± 0.98	0.00 ± 0.00	0.04 ± 0.20	31	5.84 ± 6.08	2.76 ± 0.98	0.00 ± 0.00	0.04 ± 0.20
ACS	HPR best	2.00 ± 0.00	0.00 ± 0.00	0.32 ± 0.59	50	6.82 ± 8.47	2.00 ± 0.00	0.00 ± 0.00	0.32 ± 0.59
ACS	HPR all	2.00 ± 0.00	0.00 ± 0.00	0.44 ± 0.84	50	8.30 ± 11.57	2.00 ± 0.00	0.00 ± 0.00	0.44 ± 0.84
ACS	HPP best	2.00 ± 0.00	0.00 ± 0.00	0.18 ± 0.44	50	4.92 ± 5.76	2.00 ± 0.00	0.00 ± 0.00	0.18 ± 0.44
ACS	HPP all	2.00 ± 0.00	0.00 ± 0.00	0.26 ± 0.60	50	6.00 ± 8.18	2.00 ± 0.00	0.00 ± 0.00	0.26 ± 0.60
ACS	Dynamic best	2.00 ± 0.00	0.00 ± 0.00	0.28 ± 0.54	50	6.08 ± 7.09	2.00 ± 0.00	0.00 ± 0.00	0.28 ± 0.54

Algorithm	Settings	L - Pert.	L - Confl.	L - Stag.	Opt	F - Iterations	F - Pert.	F - Confl.	F - Stag.
ACS	Dynamic all	2.00 ± 0.00	0.00 ± 0.00	0.34 ± 0.66	50	6.80 ± 8.83	2.00 ± 0.00	0.00 ± 0.00	0.34 ± 0.66
ACS	Scaled-dynamic best	2.00 ± 0.00	0.00 ± 0.00	0.34 ± 0.69	50	6.76 ± 9.00	2.00 ± 0.00	0.00 ± 0.00	0.34 ± 0.69
ACS	Scaled-dynamic all	2.00 ± 0.00	0.00 ± 0.00	0.32 ± 0.62	50	6.38 ± 8.10	2.00 ± 0.00	0.00 ± 0.00	0.32 ± 0.62
ACSFull	HPP best	0.64 ± 0.94	1.36 ± 0.94	35.02 ± 11.08	16	426.64 ± 134.88	0.64 ± 0.94	1.36 ± 0.94	35.02 ± 11.08
ACSFull	HPP all	0.52 ± 0.89	1.48 ± 0.89	35.50 ± 11.76	13	432.86 ± 143.18	0.52 ± 0.89	1.48 ± 0.89	35.50 ± 11.76
ACSFull	Dynamic best	0.32 ± 0.74	1.68 ± 0.74	37.84 ± 9.52	8	461.18 ± 115.86	0.32 ± 0.74	1.68 ± 0.74	37.84 ± 9.52
ACSFull	Dynamic all	0.44 ± 0.84	1.56 ± 0.84	35.40 ± 11.99	11	431.52 ± 146.11	0.44 ± 0.84	1.56 ± 0.84	35.40 ± 11.99
ACSFull	Scaled-dynamic best	0.40 ± 0.81	1.60 ± 0.81	37.32 ± 9.01	10	454.64 ± 110.12	0.40 ± 0.81	1.60 ± 0.81	37.32 ± 9.01
ACSFull	Scaled-dynamic all	0.48 ± 0.86	1.52 ± 0.86	36.44 ± 10.19	12	443.84 ± 124.42	0.48 ± 0.86	1.52 ± 0.86	36.44 ± 10.19
random_5x50 — optimum solution #P = 1									
Algorithm	Settings	L - Pert.	L - Confl.	L - Stag.	Opt	F - Iterations	F - Pert.	F - Confl.	F - Stag.
Online	RV PP=0	11.92 ± 2.87	0.00 ± 0.00	0.00 ± 0.00	0	1.24 ± 0.59	12.08 ± 2.97	0.00 ± 0.00	0.00 ± 0.00
Online	MC PP=0	9.24 ± 1.82	0.00 ± 0.00	0.00 ± 0.00	0	1.22 ± 0.51	11.92 ± 3.33	0.00 ± 0.00	0.00 ± 0.00
Online	RV PP=0.5	1.28 ± 0.70	0.00 ± 0.00	0.00 ± 0.00	43	1.14 ± 0.40	2.66 ± 1.06	0.00 ± 0.00	0.00 ± 0.00
Online	MC PP=0.5	1.16 ± 0.65	0.00 ± 0.00	0.00 ± 0.00	46	1.36 ± 0.69	4.64 ± 1.96	0.00 ± 0.00	0.00 ± 0.00
Online	RV PP=1	1.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	50	1.50 ± 0.81	2.08 ± 1.29	0.00 ± 0.00	0.00 ± 0.00
Online	MC PP=1	1.02 ± 0.14	0.00 ± 0.00	0.00 ± 0.00	49	1.68 ± 1.13	2.92 ± 1.63	0.00 ± 0.00	0.00 ± 0.00
Online	RV PP=1.5	1.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	50	5.50 ± 4.84	1.62 ± 0.92	0.00 ± 0.00	0.00 ± 0.00
Online	MC PP=1.5	1.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	50	2.50 ± 1.52	3.60 ± 3.06	0.00 ± 0.00	0.00 ± 0.00
ACS	HPR best	1.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	50	1.12 ± 0.33	1.48 ± 0.95	0.00 ± 0.00	0.00 ± 0.00
ACS	HPR all	1.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	50	1.16 ± 0.37	1.64 ± 0.96	0.00 ± 0.00	0.00 ± 0.00
ACS	HPP best	1.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	50	1.12 ± 0.33	1.68 ± 1.13	0.00 ± 0.00	0.00 ± 0.00
ACS	HPP all	1.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	50	1.12 ± 0.33	1.82 ± 1.22	0.00 ± 0.00	0.00 ± 0.00
ACS	Dynamic best	1.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	50	1.14 ± 0.35	2.30 ± 1.58	0.00 ± 0.00	0.00 ± 0.00
ACS	Dynamic all	1.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	50	1.16 ± 0.37	2.04 ± 1.32	0.00 ± 0.00	0.00 ± 0.00
ACS	Scaled-dynamic best	1.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	50	1.14 ± 0.35	1.62 ± 1.03	0.00 ± 0.00	0.00 ± 0.00
ACS	Scaled-dynamic all	1.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	50	1.20 ± 0.40	2.48 ± 1.59	0.00 ± 0.00	0.00 ± 0.00
ACSFull	HPP best	1.00 ± 0.00	0.00 ± 0.00	0.08 ± 0.27	50	2.46 ± 4.02	3.06 ± 1.58	0.00 ± 0.00	0.08 ± 0.27
ACSFull	HPP all	1.02 ± 0.14	0.00 ± 0.00	0.06 ± 0.24	49	1.88 ± 3.13	3.02 ± 1.66	0.00 ± 0.00	0.06 ± 0.24
ACSFull	Dynamic best	1.00 ± 0.00	0.00 ± 0.00	1.44 ± 1.73	50	19.82 ± 20.92	2.12 ± 1.45	0.00 ± 0.00	1.44 ± 1.73
ACSFull	Dynamic all	1.00 ± 0.00	0.00 ± 0.00	1.40 ± 1.51	50	18.80 ± 18.09	1.62 ± 1.09	0.00 ± 0.00	1.40 ± 1.51
ACSFull	Scaled-dynamic best	1.00 ± 0.00	0.00 ± 0.00	1.02 ± 1.50	50	14.74 ± 18.24	1.78 ± 1.02	0.00 ± 0.00	1.02 ± 1.50
ACSFull	Scaled-dynamic all	1.00 ± 0.00	0.00 ± 0.00	1.48 ± 1.43	50	19.46 ± 17.15	1.80 ± 1.11	0.00 ± 0.00	1.48 ± 1.43
random_10x50 — optimum solution #P = 3									
Algorithm	Settings	L - Pert.	L - Confl.	L - Stag.	Opt	F - Iterations	F - Pert.	F - Confl.	F - Stag.
Online	RV PP=0	29.84 ± 5.14	0.00 ± 0.00	0.00 ± 0.00	0	1.68 ± 0.84	30.76 ± 5.81	0.00 ± 0.00	0.00 ± 0.00
Online	MC PP=0	41.54 ± 22.24	0.00 ± 0.00	0.30 ± 0.65	0	15.64 ± 13.41	44.28 ± 27.47	0.00 ± 0.00	0.30 ± 0.65
Online	RV PP=0.5	5.60 ± 2.25	0.00 ± 0.00	2.14 ± 1.55	9	47.22 ± 32.66	71.04 ± 10.55	0.00 ± 0.00	2.14 ± 1.55
Online	MC PP=0.5	16.26 ± 7.04	0.00 ± 0.00	2.12 ± 1.36	1	48.80 ± 27.85	89.38 ± 6.98	0.00 ± 0.00	2.12 ± 1.36
Online	RV PP=1	0.88 ± 2.02	1.64 ± 0.78	21.66 ± 4.18	1	469.18 ± 91.85	1.78 ± 4.13	1.64 ± 0.78	21.66 ± 4.18
Online	MC PP=1	5.40 ± 4.70	0.20 ± 0.61	7.06 ± 6.79	0	159.68 ± 151.39	20.32 ± 8.96	0.20 ± 0.61	7.06 ± 6.79
Online	RV PP=1.5	0.00 ± 0.00	2.00 ± 0.00	23.00 ± 0.00	0	500.00 ± 0.00	0.00 ± 0.00	2.00 ± 0.00	23.00 ± 0.00
Online	MC PP=1.5	0.80 ± 1.93	1.68 ± 0.74	21.20 ± 4.92	0	460.20 ± 107.42	1.30 ± 3.15	1.68 ± 0.74	21.20 ± 4.92
ACS	HPR best	3.00 ± 0.00	0.00 ± 0.00	0.80 ± 1.03	50	13.56 ± 14.27	5.28 ± 1.83	0.00 ± 0.00	0.80 ± 1.03
ACS	HPR all	3.00 ± 0.00	0.00 ± 0.00	1.32 ± 1.72	50	20.60 ± 23.42	5.20 ± 2.12	0.00 ± 0.00	1.32 ± 1.72
ACS	HPP best	3.18 ± 0.39	0.00 ± 0.00	0.80 ± 1.03	41	13.32 ± 14.68	7.18 ± 3.26	0.00 ± 0.00	0.80 ± 1.03
ACS	HPP all	3.32 ± 0.47	0.00 ± 0.00	0.68 ± 0.96	34	11.22 ± 12.76	6.54 ± 2.35	0.00 ± 0.00	0.68 ± 0.96
ACS	Dynamic best	3.40 ± 0.53	0.00 ± 0.00	0.80 ± 1.11	31	13.82 ± 15.44	7.68 ± 2.74	0.00 ± 0.00	0.80 ± 1.11
ACS	Dynamic all	3.18 ± 0.39	0.00 ± 0.00	0.88 ± 1.15	41	15.02 ± 16.57	7.22 ± 2.64	0.00 ± 0.00	0.88 ± 1.15

Algorithm	Settings	L - Pert.	L - Confl.	L - Stag.	Opt	F - Iterations	F - Pert.	F - Confl.	F - Stag.
ACS	Scaled-dynamic best	3.34 ± 0.56	0.00 ± 0.00	0.60 ± 0.99	35	10.64 ± 13.91	7.74 ± 2.86	0.00 ± 0.00	0.60 ± 0.99
ACS	Scaled-dynamic all	3.46 ± 0.61	0.00 ± 0.00	0.66 ± 1.00	30	11.40 ± 13.87	8.44 ± 3.94	0.00 ± 0.00	0.66 ± 1.00
ACSFull	HPP best	9.22 ± 2.29	0.00 ± 0.00	3.08 ± 3.30	0	52.36 ± 52.39	16.22 ± 4.74	0.00 ± 0.00	3.08 ± 3.30
ACSFull	HPP all	8.84 ± 2.24	0.00 ± 0.00	3.12 ± 3.29	0	51.80 ± 49.80	15.90 ± 4.68	0.00 ± 0.00	3.12 ± 3.29
ACSFull	Dynamic best	6.36 ± 2.42	0.04 ± 0.28	10.48 ± 9.44	4	169.88 ± 145.00	8.22 ± 2.87	0.04 ± 0.28	10.48 ± 9.44
ACSFull	Dynamic all	5.34 ± 2.80	0.24 ± 0.66	12.74 ± 10.60	5	197.08 ± 161.59	6.98 ± 3.78	0.24 ± 0.66	12.74 ± 10.60
ACSFull	Scaled-dynamic best	5.16 ± 2.92	0.20 ± 0.61	12.72 ± 10.00	10	204.08 ± 157.16	7.30 ± 4.34	0.20 ± 0.61	12.72 ± 10.00
ACSFull	Scaled-dynamic all	6.32 ± 3.32	0.12 ± 0.48	11.80 ± 9.97	3	180.28 ± 151.06	7.18 ± 3.53	0.12 ± 0.48	11.80 ± 9.97
random_20x50 — optimum solution #P = 4									
Algorithm	Settings	L - Pert.	L - Confl.	L - Stag.	Opt	F - Iterations	F - Pert.	F - Confl.	F - Stag.
Online	RV PP=0	69.22 ± 5.00	0.16 ± 0.55	1.90 ± 6.29	0	45.66 ± 135.61	71.34 ± 5.32	0.16 ± 0.55	1.90 ± 6.29
Online	MC PP=0	57.18 ± 6.15	2.24 ± 0.66	22.62 ± 0.53	0	500.00 ± 0.00	57.18 ± 6.15	2.24 ± 0.66	22.62 ± 0.53
Online	RV PP=0.5	1.28 ± 9.05	1.96 ± 0.28	22.86 ± 0.99	0	496.76 ± 22.91	3.94 ± 27.86	1.96 ± 0.28	22.86 ± 0.99
Online	MC PP=0.5	0.48 ± 0.81	2.00 ± 0.00	22.94 ± 0.24	0	500.00 ± 0.00	0.48 ± 0.81	2.00 ± 0.00	22.94 ± 0.24
Online	RV PP=1	0.16 ± 0.79	1.92 ± 0.40	22.44 ± 3.02	2	487.64 ± 66.25	0.50 ± 2.48	1.92 ± 0.40	22.44 ± 3.02
Online	MC PP=1	0.28 ± 0.83	2.00 ± 0.00	22.96 ± 0.20	0	500.00 ± 0.00	0.28 ± 0.83	2.00 ± 0.00	22.96 ± 0.20
Online	RV PP=1.5	0.00 ± 0.00	2.00 ± 0.00	23.00 ± 0.00	0	500.00 ± 0.00	0.00 ± 0.00	2.00 ± 0.00	23.00 ± 0.00
Online	MC PP=1.5	0.66 ± 2.62	1.84 ± 0.55	21.84 ± 4.73	2	474.74 ± 102.27	1.44 ± 5.08	1.84 ± 0.55	21.84 ± 4.73
ACS	HPR best	5.34 ± 1.84	0.16 ± 0.55	7.54 ± 8.33	9	132.40 ± 138.12	10.62 ± 5.07	0.16 ± 0.55	7.54 ± 8.33
ACS	HPR all	5.00 ± 1.87	0.12 ± 0.48	6.62 ± 8.41	22	115.46 ± 141.21	9.12 ± 3.50	0.12 ± 0.48	6.62 ± 8.41
ACS	HPP best	13.82 ± 6.13	0.04 ± 0.28	7.78 ± 6.92	0	154.62 ± 130.91	20.04 ± 7.28	0.04 ± 0.28	7.78 ± 6.92
ACS	HPP all	12.96 ± 5.84	0.08 ± 0.40	7.60 ± 6.72	0	152.74 ± 132.69	20.52 ± 7.26	0.08 ± 0.40	7.60 ± 6.72
ACS	Dynamic best	21.54 ± 8.79	0.16 ± 0.55	9.18 ± 7.88	0	189.86 ± 155.71	30.76 ± 10.91	0.16 ± 0.55	9.18 ± 7.88
ACS	Dynamic all	18.86 ± 7.74	0.20 ± 0.61	8.06 ± 7.93	0	170.10 ± 160.42	29.96 ± 11.15	0.20 ± 0.61	8.06 ± 7.93
ACS	Scaled-dynamic best	24.38 ± 7.38	0.40 ± 0.81	8.52 ± 8.87	0	179.22 ± 177.28	31.54 ± 9.75	0.40 ± 0.81	8.52 ± 8.87
ACS	Scaled-dynamic all	20.60 ± 7.71	0.12 ± 0.48	7.10 ± 7.43	0	150.48 ± 151.88	32.66 ± 11.25	0.12 ± 0.48	7.10 ± 7.43
ACSFull	HPP best	48.70 ± 8.20	2.26 ± 0.63	13.52 ± 1.80	0	493.52 ± 45.82	48.76 ± 8.22	2.26 ± 0.63	13.52 ± 1.80
ACSFull	HPP all	44.42 ± 9.38	2.66 ± 0.92	18.64 ± 1.31	0	495.70 ± 30.41	44.46 ± 9.40	2.66 ± 0.92	18.64 ± 1.31
ACSFull	Dynamic best	21.54 ± 10.29	2.16 ± 0.62	13.40 ± 1.28	0	494.64 ± 37.90	21.54 ± 10.29	2.16 ± 0.62	13.40 ± 1.28
ACSFull	Dynamic all	16.18 ± 6.77	2.46 ± 0.79	18.74 ± 0.60	0	500.00 ± 0.00	16.18 ± 6.77	2.46 ± 0.79	18.74 ± 0.60
ACSFull	Scaled-dynamic best	19.66 ± 6.77	2.08 ± 0.40	13.48 ± 0.79	0	500.00 ± 0.00	19.66 ± 6.77	2.08 ± 0.40	13.48 ± 0.79
ACSFull	Scaled-dynamic all	17.88 ± 7.43	2.54 ± 0.84	18.70 ± 0.61	0	500.00 ± 0.00	17.88 ± 7.43	2.54 ± 0.84	18.70 ± 0.61
random_10x100 — optimum solution #P = 7									
Algorithm	Settings	L - Pert.	L - Confl.	L - Stag.	Opt	F - Iterations	F - Pert.	F - Confl.	F - Stag.
Online	RV PP=0	123.90 ± 32.66	0.00 ± 0.00	0.10 ± 0.36	0	3.88 ± 7.98	124.34 ± 33.92	0.00 ± 0.00	0.10 ± 0.36
Online	MC PP=0	222.84 ± 8.37	0.00 ± 0.00	1.48 ± 0.91	0	34.04 ± 19.30	228.30 ± 9.43	0.00 ± 0.00	1.48 ± 0.91
Online	RV PP=0.5	10.62 ± 3.16	0.00 ± 0.00	1.02 ± 0.43	6	30.28 ± 15.50	74.58 ± 26.68	0.00 ± 0.00	1.02 ± 0.43
Online	MC PP=0.5	9.58 ± 6.01	0.00 ± 0.00	0.08 ± 0.27	1	4.72 ± 8.02	21.92 ± 32.12	0.00 ± 0.00	0.08 ± 0.27
Online	RV PP=1	3.16 ± 3.86	2.70 ± 0.91	7.56 ± 6.33	0	477.96 ± 86.67	4.10 ± 6.52	2.70 ± 0.91	7.56 ± 6.33
Online	MC PP=1	10.34 ± 5.70	0.78 ± 1.33	2.52 ± 2.34	0	191.72 ± 191.00	35.34 ± 21.42	0.78 ± 1.33	2.52 ± 2.34
Online	RV PP=1.5	2.00 ± 0.00	3.00 ± 0.00	22.64 ± 0.48	0	500.00 ± 0.00	2.00 ± 0.00	3.00 ± 0.00	22.64 ± 0.48
Online	MC PP=1.5	3.22 ± 1.62	3.00 ± 0.64	7.48 ± 7.52	0	500.00 ± 0.00	3.22 ± 1.62	3.00 ± 0.64	7.48 ± 7.52
ACS	HPR best	8.92 ± 0.53	0.00 ± 0.00	0.58 ± 1.23	2	12.48 ± 17.72	11.96 ± 1.89	0.00 ± 0.00	0.58 ± 1.23
ACS	HPR all	8.74 ± 0.49	0.00 ± 0.00	0.46 ± 0.99	1	10.96 ± 14.90	11.38 ± 1.87	0.00 ± 0.00	0.46 ± 0.99
ACS	HPP best	8.32 ± 0.89	0.00 ± 0.00	0.12 ± 0.39	13	5.78 ± 5.41	11.38 ± 1.95	0.00 ± 0.00	0.12 ± 0.39
ACS	HPP all	8.36 ± 0.80	0.00 ± 0.00	0.16 ± 0.42	10	6.12 ± 5.92	11.70 ± 1.83	0.00 ± 0.00	0.16 ± 0.42
ACS	Dynamic best	8.08 ± 0.90	0.00 ± 0.00	0.26 ± 0.53	18	7.76 ± 7.99	12.54 ± 2.48	0.00 ± 0.00	0.26 ± 0.53
ACS	Dynamic all	7.90 ± 0.89	0.00 ± 0.00	0.22 ± 0.42	22	7.32 ± 6.84	12.72 ± 3.45	0.00 ± 0.00	0.22 ± 0.42
ACS	Scaled-dynamic best	8.36 ± 1.08	0.00 ± 0.00	0.12 ± 0.48	15	6.66 ± 8.06	13.14 ± 3.18	0.00 ± 0.00	0.12 ± 0.48

Algorithm	Settings	L - Pert.	L - Confl.	L - Stag.	Opt	F - Iterations	F - Pert.	F - Confl.	F - Stag.
ACS	Scaled-dynamic all	8.08 ± 0.90	0.00 ± 0.00	0.16 ± 0.42	17	6.00 ± 5.96	12.20 ± 2.50	0.00 ± 0.00	0.16 ± 0.42
ACSFull	HPP best	20.98 ± 4.44	0.00 ± 0.00	5.76 ± 5.81	0	105.46 ± 103.65	27.08 ± 4.57	0.00 ± 0.00	5.76 ± 5.81
ACSFull	HPP all	20.62 ± 5.29	0.16 ± 0.55	10.72 ± 10.31	0	177.84 ± 167.71	22.64 ± 5.52	0.16 ± 0.55	10.72 ± 10.31
ACSFull	Dynamic best	11.08 ± 5.06	1.38 ± 1.03	23.42 ± 6.87	0	428.40 ± 126.89	11.34 ± 5.26	1.38 ± 1.03	23.42 ± 6.87
ACSFull	Dynamic all	9.46 ± 4.20	1.62 ± 0.90	25.92 ± 8.93	0	431.34 ± 146.87	9.72 ± 4.59	1.62 ± 0.90	25.92 ± 8.93
ACSFull	Scaled-dynamic best	10.48 ± 5.09	1.38 ± 1.03	23.82 ± 6.49	0	432.72 ± 117.09	11.24 ± 5.99	1.38 ± 1.03	23.82 ± 6.49
ACSFull	Scaled-dynamic all	9.52 ± 3.73	1.60 ± 0.95	26.10 ± 8.83	0	431.58 ± 145.96	9.68 ± 3.91	1.60 ± 0.95	26.10 ± 8.83
random_10x200 — optimum solution #P = 12									
Algorithm	Settings	L - Pert.	L - Confl.	L - Stag.	Opt	F - Iterations	F - Pert.	F - Confl.	F - Stag.
Online	RV PP=0	250.04 ± 8.84	0.00 ± 0.00	0.00 ± 0.00	0	1.00 ± 0.00	250.04 ± 8.84	0.00 ± 0.00	0.00 ± 0.00
Online	MC PP=0	163.66 ± 23.62	0.00 ± 0.00	0.00 ± 0.00	0	1.88 ± 1.35	163.66 ± 23.62	0.00 ± 0.00	0.00 ± 0.00
Online	RV PP=0.5	24.12 ± 9.90	0.74 ± 1.43	1.48 ± 0.91	0	144.84 ± 191.79	137.94 ± 70.69	0.74 ± 1.43	1.48 ± 0.91
Online	MC PP=0.5	87.08 ± 41.64	0.78 ± 1.50	1.06 ± 0.31	0	139.72 ± 193.65	189.92 ± 96.78	0.78 ± 1.50	1.06 ± 0.31
Online	RV PP=1	9.02 ± 4.31	3.06 ± 1.79	9.88 ± 8.18	0	428.04 ± 143.64	14.24 ± 14.89	3.06 ± 1.79	9.88 ± 8.18
Online	MC PP=1	26.60 ± 13.98	1.52 ± 2.36	2.44 ± 2.96	0	207.04 ± 206.11	64.52 ± 41.23	1.52 ± 2.36	2.44 ± 2.96
Online	RV PP=1.5	2.78 ± 0.65	7.24 ± 0.62	21.60 ± 3.64	0	500.00 ± 0.00	2.78 ± 0.65	7.24 ± 0.62	21.60 ± 3.64
Online	MC PP=1.5	3.92 ± 0.80	6.16 ± 0.55	19.38 ± 7.11	0	500.00 ± 0.00	3.92 ± 0.80	6.16 ± 0.55	19.38 ± 7.11
ACS	HPR best	13.52 ± 1.13	0.00 ± 0.00	2.26 ± 2.76	9	42.72 ± 44.10	18.28 ± 2.81	0.00 ± 0.00	2.26 ± 2.76
ACS	HPR all	13.40 ± 1.05	0.00 ± 0.00	2.30 ± 3.27	11	44.50 ± 53.49	17.38 ± 2.64	0.00 ± 0.00	2.30 ± 3.27
ACS	HPP best	14.16 ± 1.23	0.00 ± 0.00	0.70 ± 0.89	8	16.44 ± 13.47	18.02 ± 2.21	0.00 ± 0.00	0.70 ± 0.89
ACS	HPP all	14.30 ± 1.22	0.00 ± 0.00	0.56 ± 1.11	8	13.88 ± 16.65	18.04 ± 2.39	0.00 ± 0.00	0.56 ± 1.11
ACS	Dynamic best	14.54 ± 0.91	0.00 ± 0.00	0.38 ± 0.60	3	10.94 ± 9.38	17.68 ± 2.30	0.00 ± 0.00	0.38 ± 0.60
ACS	Dynamic all	14.08 ± 1.16	0.00 ± 0.00	0.22 ± 0.55	6	8.60 ± 8.10	17.96 ± 2.90	0.00 ± 0.00	0.22 ± 0.55
ACS	Scaled-dynamic best	14.46 ± 1.03	0.00 ± 0.00	0.24 ± 0.59	4	8.78 ± 9.67	18.34 ± 2.54	0.00 ± 0.00	0.24 ± 0.59
ACS	Scaled-dynamic all	14.66 ± 0.87	0.00 ± 0.00	0.46 ± 0.61	1	11.94 ± 10.16	18.80 ± 2.66	0.00 ± 0.00	0.46 ± 0.61
ACSFull	HPP best	27.26 ± 1.86	0.00 ± 0.00	0.84 ± 1.27	0	27.26 ± 30.98	35.00 ± 4.94	0.00 ± 0.00	0.84 ± 1.27
ACSFull	HPP all	28.00 ± 2.46	0.00 ± 0.00	2.02 ± 1.92	0	48.00 ± 38.13	34.60 ± 5.79	0.00 ± 0.00	2.02 ± 1.92
ACSFull	Dynamic best	20.34 ± 2.85	0.00 ± 0.00	4.26 ± 4.71	0	107.26 ± 106.15	25.26 ± 3.77	0.00 ± 0.00	4.26 ± 4.71
ACSFull	Dynamic all	19.78 ± 4.97	0.48 ± 0.86	12.22 ± 9.26	0	257.66 ± 189.02	20.78 ± 5.49	0.48 ± 0.86	12.22 ± 9.26
ACSFull	Scaled-dynamic best	20.40 ± 2.44	0.00 ± 0.00	5.40 ± 5.45	0	134.02 ± 125.99	23.86 ± 3.02	0.00 ± 0.00	5.40 ± 5.45
ACSFull	Scaled-dynamic all	19.48 ± 5.63	0.56 ± 0.91	14.74 ± 8.29	0	307.96 ± 167.65	20.72 ± 6.13	0.56 ± 0.91	14.74 ± 8.29