Comparison of Naive and UNSAT Core giuded Paths Search

This document contains the full description of the problem instances used to compare the naive and UNSAT Core guided path search for alternative conflict-free paths.

Comparing the improved algorithm to the original algorithm

First of all, three instances of increasing difficulty are examined in order to learn and understand the workings of the new capacity verification problem and new paths changing problem. In these instances one can observe the value of the improved algorithm very well.

Instance 1

Consider the instance defined in Table 1. In this instance there are two jobs, which both have a starting location, one task, and an ending location. The time windows of both tasks are identical and equal to [11,13]. Job 1 needs to go from node 0 to node 1, perform its task, and then go back. Additionally, job 2 needs to go from node 1 to node 0, perform its task, and go back. Node 0 and node 1 are directly connected to each via an edge. Furthermore, there exists another node, namely node 2, which indirectly connects node 0 and 1. Lastly, because node 0 and 1 are nodes where routes start and end, these two nodes are depots, which are locations where an infinite number of vehicles can be present.

Table 1: Jobs information of instance 1

Job	Task	Location	Processing time	Time window
1	**Start**	0	0	-
	0	1	5	[11,13]
	End	0	0	-
2	**Start**	1	0	-
	1	0	6	[11,13]
	End	1	0	-

Figure 1 illustrates the initial solution to the problem. Additionally, in Appendix the exact nodes and edges that are used can be found. In the initial solution, the paths changing problem has not been invoked yet.

The initial solution turns out to be unsatisfiable. As can be seen, the path from 0 via 2 to 1 has a length of 12, and obviously so does the path from 1 via 2 to 0. In addition, the path from 0 to 1 and from 1 to 0 has a length of 10. Logically, because the objective function is to minimize the traveled distance, both jobs try to use the shortest path, which is consists of edges (0,1) and (1,0) for job 1 and edges (1,0) and (0,1) for job 2. It should be noted that only the first time that jobs 1 and 2 cross their paths, a conflict occurs because both jobs want to use the same edge at the same time. Although Figure 1 might suggest that the jobs cross a second time, this does not happen in reality. In reality, a vehicle corresponding to one of the jobs waits on its node before it starts travelling back, because there are no time windows for being at the end, which allows for this behavior. This is only possible because both node 0 and 1 are depots.

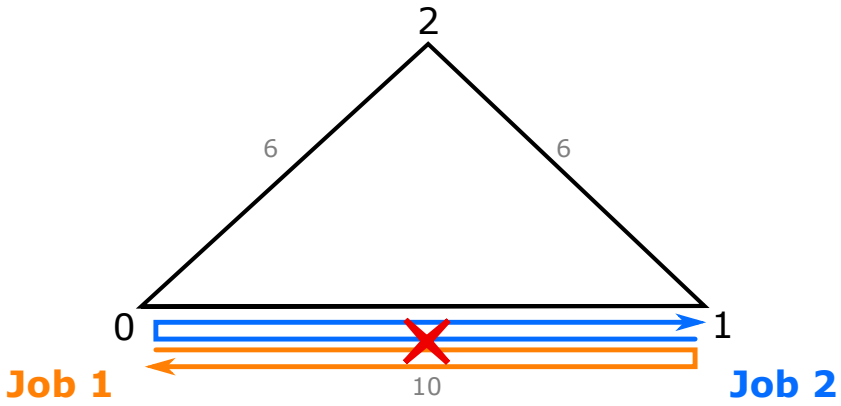


Figure 1: Initial solution of Instance 1.

In this instance only one unsatisfiable capacity constraint is filtered out:

$$\text{Capacity edge - opposite: R1 P(0,1) E(0,1) - R2 P(1,0) E(1,0)} \tag{0.0.1}$$

So, at most one of route 1 - pair (0,1) and route 2 - pair (1,0) can use edge (0,1) (which is identical to (1,0), except in the opposite direction), otherwise the time windows of the tasks cannot be satisfied. After one iteration, the improved paths changing problem finds the solution in Figure 2, which is a satisfiable solution. As can be seen, route 2 - pair (1,0) no longer uses edge (1,0), but has changed its path and goes from 1 to 0 through 2. Note that the time window of job 2 task 1 is still satisfied, because the travel time is 12 and the time window [11,13].

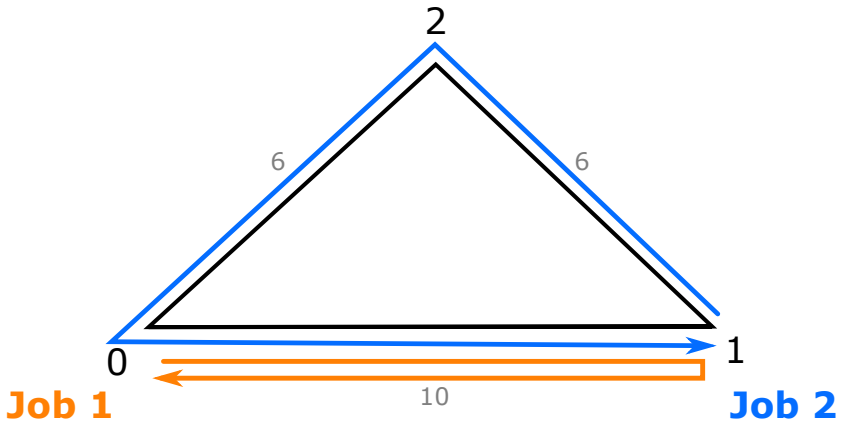
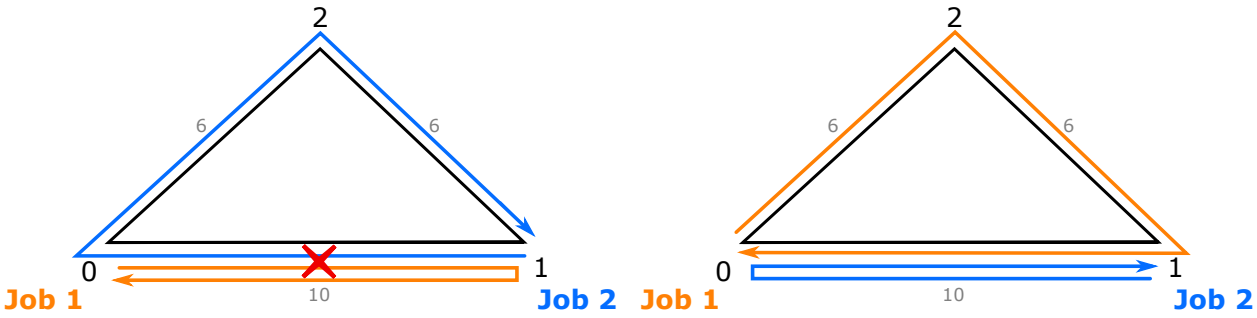


Figure 2: Improved paths changer: New path attempt 1.

The original paths changing problem, on the other hand, does not find a satisfiable solution straight away. The first attempt can be seen in Figure 3a. Here, neither route 1 - pair (0,1) nor route 2 - pair (1,0) has been changed, which were the two conflicting combinations of routes and pairs. Rather, the path of route 2 - pair (0,1) has been changed, but this is not one of the pairs that caused unsatisfiability. In this attempt, the paths changer just arbitrarily chose a different edge in an attempt of finding a feasible solution.

After one more iteration, the original paths changing problem also finds a feasible solution, as shown in Figure 3b. The original and improved paths changing problem do not find the same results. However, they do find a solution with an equal traveled distance.



(a) Original paths changer: New path attempt 1

(b) Original paths changer: New path attempt 2

Figure 3: New paths generated by the original paths changer.

This instance only scratches the surface of the value of the improved paths changer. In this instance, the computation time of the entire algorithm is decreased from $0.250\,[s]$ for the original paths changer to $0.161\,[s]$ for the improved paths changer. From these results one might (undeservedly) conclude that it is not really worth the effort of adjusting the original algorithm, in order to decrease the the computation time by only $0.089\,[s]$. However, this is only the smallest possible instance, where in total just 8 paths are possible.

Instance 2

Now, let us look a slightly more complicated instance as defined in Table 2. In this instance three jobs are defined, where job 1 and job 2 are identical. Jobs 1 and 2 both start at node 0, visit node 3 to perform their tasks, and then go back. On the other hand, job 3 starts at node 3, visits node 0 to perform its task, and returns to node 3. Additionally, all tasks have identical time windows, which are $[17, 19]$. The network of nodes and edges, as well as the unsatisfiable initial solution, can be seen in Figure 4. Here, node 0 and node 3 are depots with infinite capacity. Furthermore, the list of the exact nodes and edges that are used can be found in Appendix .

Table 2: Jobs information of Instance 2

Job	Task	Location	Processing time	Time window
1	**Start**	0	0	-
	1	3	5	[17,19]
	End	0	0	-
2	**Start**	0	0	-
	1	3	5	[17,19]
	End	0	0	-
3	**Start**	3	0	-
	1	0	3	[17,19]
	End	3	0	-

In the initial solution, all jobs try to use the shortest possible path, which is straight from node 0 through node 1 and 2 to node 3 for jobs 1 and 2, and from node 3 through node 2 and 1 to node 0 for job 3. Because of the time windows on the tasks, it is not possible for conflicting vehicles to wait for each other on the way to their respective tasks. For this reason they they have to be rerouted. However, because vehicles can wait at depots before they transit to their end locations, the way back is irrelevant. This is only possible because there are no time windows for being at the end locations.

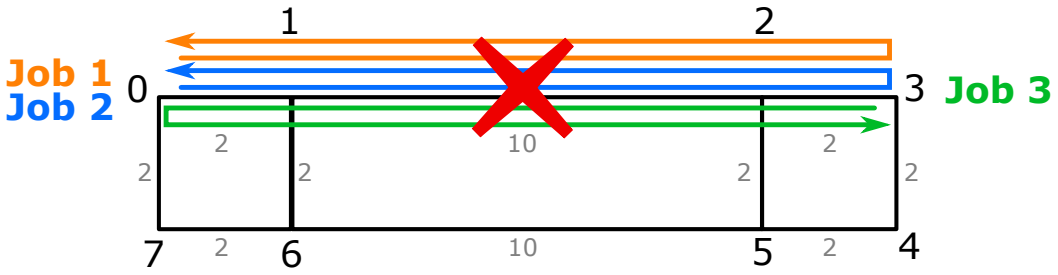


Figure 4: Initial solution of Instance 2.

This instance has two unsatisfiable capacity constraints as a result of the time windows of the tasks, making up two unsatisfiable cores, namely:

$$\begin{aligned} &\text{Capacity edge - opposite: R1 P(0,3) E(1,2) - R3 P(3,0) E(2,1)} \\ &\text{Capacity edge - opposite: R2 P(0,3) E(1,2) - R3 P(3,0) E(2,1)} \end{aligned} \tag{0.0.2}$$

As can be seen, route 1 - pair (0,3) and route 2 - pair (0,3) are both in conflict with the same route, namely route 3 - pair (3,0). This is so, since route 1 and route 2 are identical. From the first unsatisfiable constraint it is derived that either route 1 - pair (0,3) or route 3 - pair (3,0) cannot use edge (1,2) (or (2,1), which is the same edge but in the opposite direction). From the second unsatisfiable constraint it is derived that either route 2 - pair (0,3) or route 3 - pair (3,0) cannot use edge (1,2) (or (2,1), which is again the same edge). After one iteration, the improved paths changing problem finds the satisfiable solution shown in Figure 5. Since the traveled distance is minimized, it makes sense that route 3 - pair (3,0) is assigned a different path. If route 3 - pair (3,0) would still use edge (2,1), this would mean that both route 1 - pair (0,3) and route 2 - pair (0,3) would have to be rerouted, which would lead to a larger traveled distance.

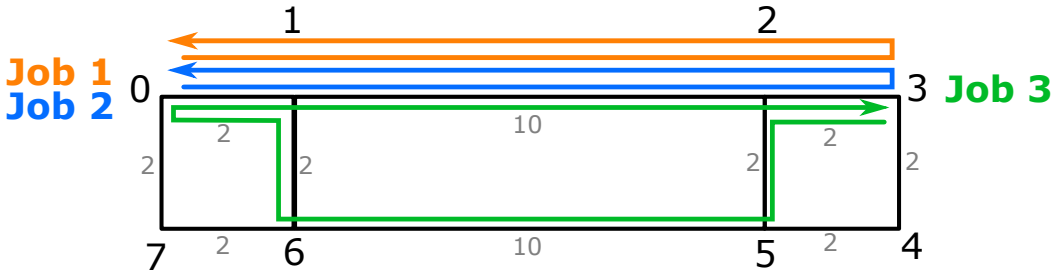
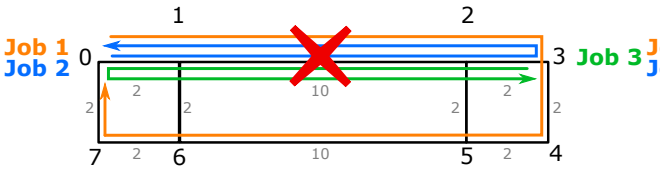
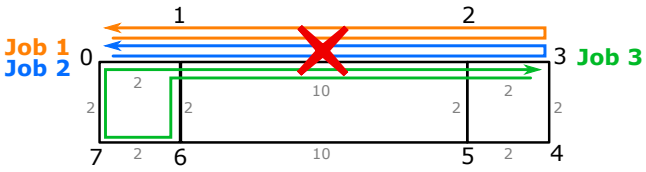


Figure 5: Improved paths changer: New path attempt 1.

Figures 6a and 6b show the first two iterations of the original paths changing problem. This instance shows very well how the original paths changing problem behaves. In Figure 6a one can observe that route 1 has been adjusted by the paths changer, but again the adjustment does not appear in one of the pairs that is in conflict. Route 1 - pair (0,3) is untouched, and instead route 1 - pair (3,0) is changed. Moreover, even if this would resolve the conflict between route 1 - pair (0,3) and route 3 - pair (3,0), then route 2 - pair (0,3) and route 3 - pair (3,0) would still be in conflict.



(a) Original paths changer: New path attempt 1



(b) Original paths changer: New path attempt 2

Figure 6: New paths generated by the original paths changer.

Finally, after 24 iterations, the original paths changer manages to find a feasible solution, shown in Figure 7. In this solution, one of the conflicting combinations of routes and pairs has found a different path, namely route 3 - pair (3,0). Because of this, route 1 - pair (0,3) and route 2 - pair (0,3) can still use their original path. Again, the original and improved paths changer problem do not find the exact same solution, but the traveled distance is the same.

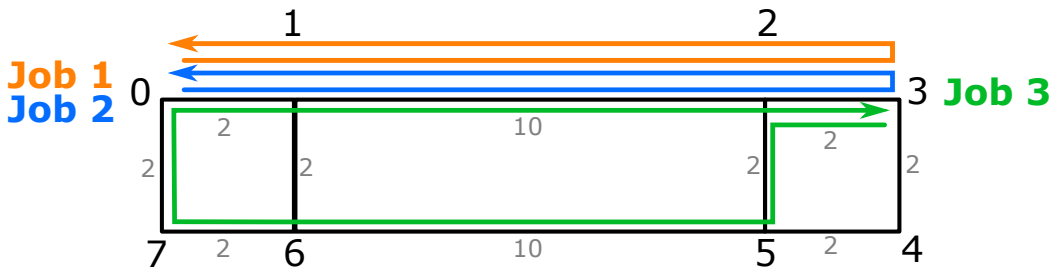


Figure 7: Original paths changer: New path attempt 24.

In this instance the computation time of the entire algorithm has been reduced from 8.805 $[s]$ for the original paths changer to 0.467 $[s]$ for the improved paths changer. This is already a more significant improvement in computation time in comparison to Instance 1.

Instance 3

Lastly, an instance will be introduced that is even more computationally demanding. In Table 3, a set of four jobs has been defined. Job 1 starts at node 0, visits node 2 to perform its task, and returns to node 0; job 2 starts at node 1, visits node 3 to perform its task, and goes back to node 1; job 3 starts at node 2, visits node 0 to perform its task, and goes back to node 2; and lastly job 4 starts at node 3, visits node 1 to perform its task, and returns to node 3. All of the tasks have a time window of $[19, 22]$. The network of nodes and edges, and the initial solution, can be found in Figure 8. In this network, node 0, 1, 2 and 3 are all depots with infinite capacity. Appendix lists the exact nodes and edges that are used for this instance.

Table 3: Jobs information of Instance 3

Job	Task	Location	Processing time	Time window
1	**Start**	0	0	-
	1	2	5	[19,22]
	End	0	0	-
2	**Start**	1	0	-
	1	3	3	[19,22]
	End	1	0	-
3	**Start**	2	0	-
	1	0	5	[19,22]
	End	2	0	-
4	**Start**	3	0	-
	1	1	6	[19,22]
	End	3	0	-

Because all vehicles have to go to the opposite side of where they start in order to perform their tasks, minimizing the traveled distance leads to a solution where all vehicles prefer to go through node 4, as can be recognized in Figure 8. The diagonal distance is namely 14, whereas the path over the edges of the square is equal to 20 (10 per edge). Also in this instance, the paths on the way back are not in conflict because there are no time windows for being at the end locations, so vehicles can wait at depots before returning.

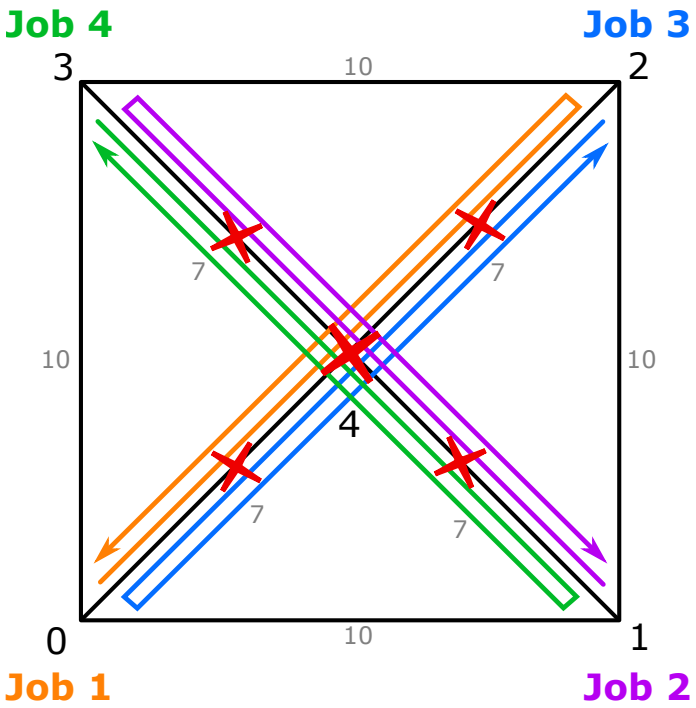


Figure 8: Initial solution of Instance 3.

This instance has a total of six unsatisfiable capacity constraints, resulting from the time windows of the tasks, namely:

Capacity node: R1 P(0,2) N4 - R3 P(2,0) N4
Capacity edge - opposite: R1 P(0,2) E(0,4) - R3 P(2,0) E(4,0)
Capacity edge - opposite: R1 P(0,2) E(4,2) - R3 P(2,0) E(2,4)
Capacity node: R2 P(1,3) N4 - R4 P(3,1) N4
Capacity edge - opposite: R2 P(1,3) E(1,4) - R4 P(3,1) E(4,1)
Capacity edge - opposite: R2 P(1,3) E(4,3) - R4 P(3,1) E(3,4)
(0.0.3)

These unsatisfiable constraints are divided over two separate unsatisfiable cores. The first unsatisfiable core is the conflict between route 1 - pair (0,2) and route 3 - pair (2,0). These routes are conflicting at node 4 and edge (0,4) and (4,2) (or edge (4,0) and (2,4), depending on which direction the vehicle travels in). The second unsatisfiable core is the conflict between route 2 - pair (1,3) and route 4 - pair (3,1), which are conflicting at node 4 and edge (1,4) and (4,3) (or edge (4,1) and (3,4), depending on which direction the vehicle travels in).

The improved paths changer finds a feasible solution to this problem after only one iteration. This solution can be seen in Figure 9. As can be seen, routes 1 and 2 are untouched. However, the path of pair (2,0) has been changed for route 3. Similarly, the path of pair (3,1) has been changed for route 4.

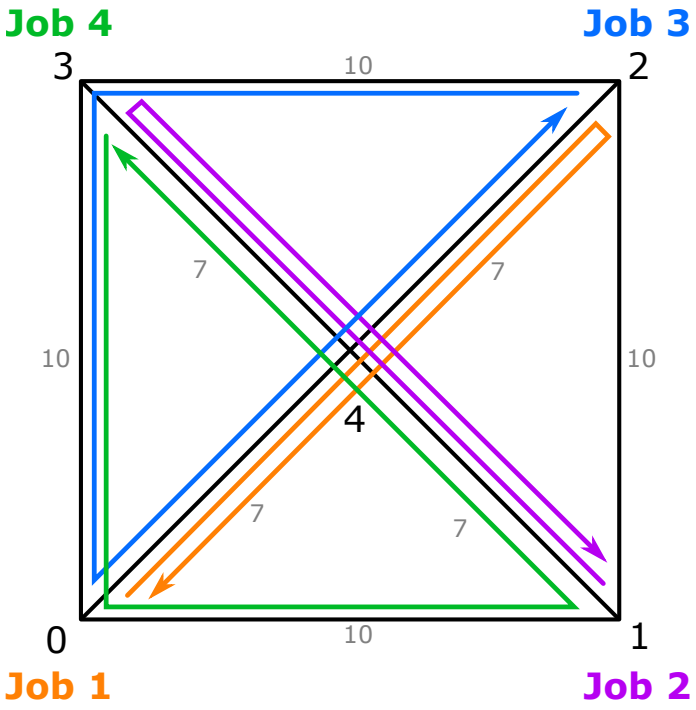
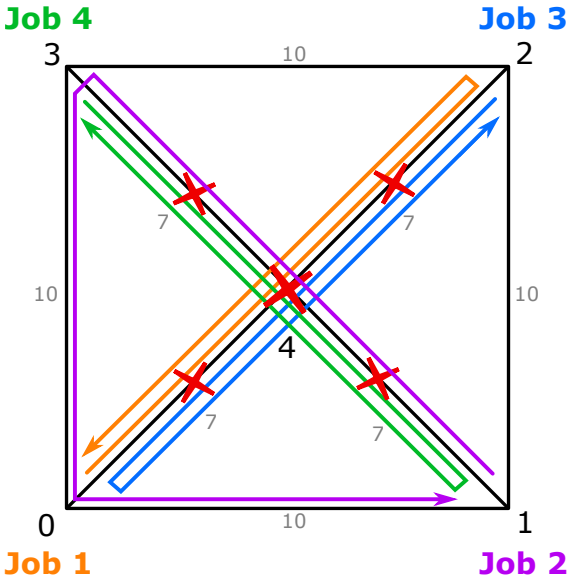
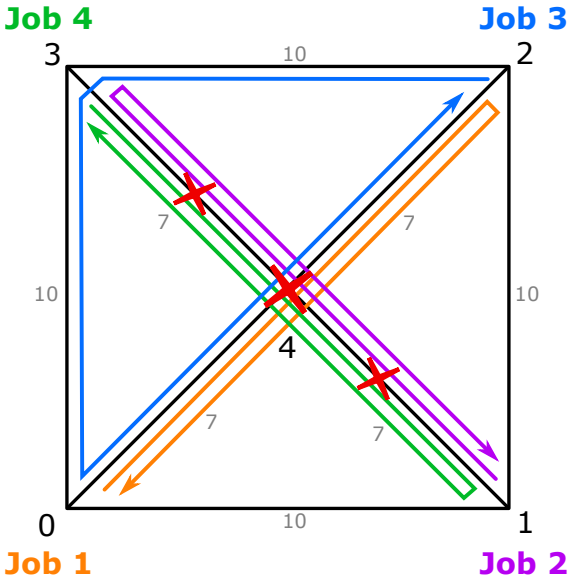


Figure 9: Improved paths changer: New path attempt 1.

In contrast, the original paths changer does not find a feasible solution so fast. Figures 10a and 10b show the first and second iteration of the original paths changer. For the first iteration, all previous conflicts are still present, because a pair that was actually satisfied has been changed. For the second iteration, half of the conflicts have been solved, because route 3 - pair (2,0) has been rerouted. However, this is more a coincidence than a deliberate decision, as will be explained next.



(a) Original paths changer: New path attempt 1



(b) Original paths changer: New path attempt 2

Figure 10: New paths generated by the original paths changer.

Because half of the conflicts have been resolved while going from attempt 1 to attempt 2, one might conclude that the algorithm is 'moving in the right direction'. However, the original algorithm has no idea that this solution is a step in the right direction, and the next attempt is worse than the previous one. Figure 11 shows attempt 3 of the original paths changer. Here, it can be seen that all conflicts of (0.0.3) are present again and nothing has been resolved. The change that had been made to route 3 - pair (2,0) in attempt 2 has been undone, even though it was a positive change.

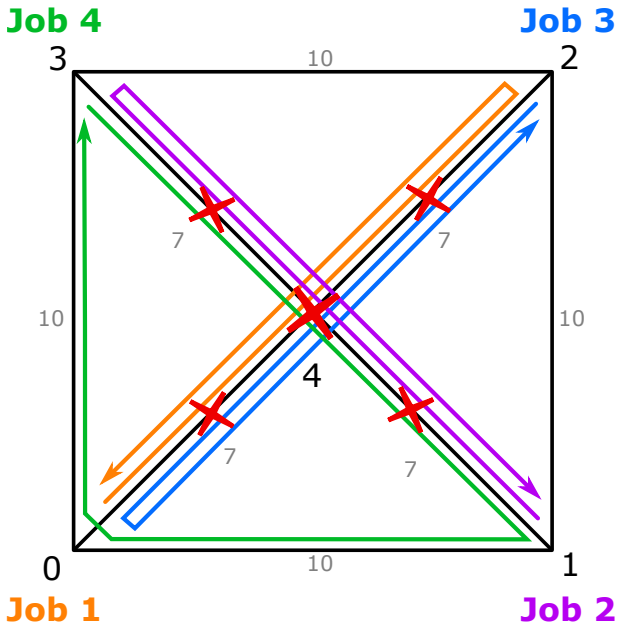


Figure 11: Original paths changer: New path attempt 3.

Finally, after 54 iterations, the original paths changer finds a feasible solution, shown in Figure 12. Similarly to the previous two instances, the original and improved paths changer find different solutions, but the traveled distance is the same.

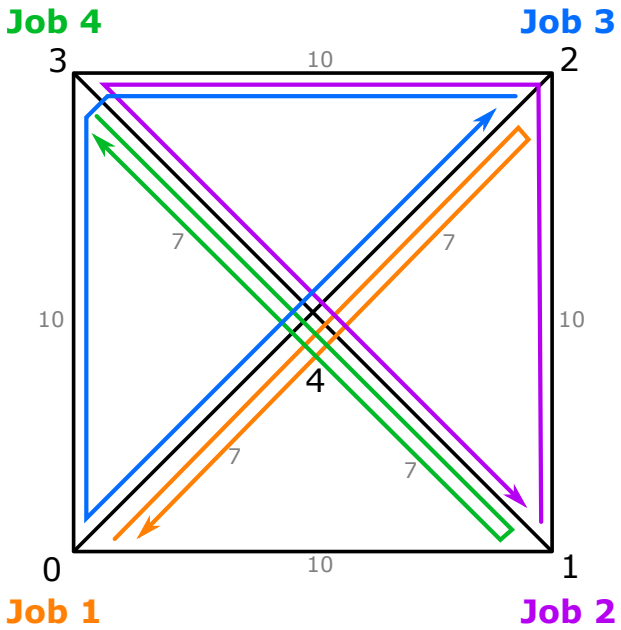


Figure 12: Original paths changer: New path attempt 54.

For the final instance, the computation time of the entire algorithm has been reduced from 35.923 $[s]$ for the original paths changer to 1.083 $[s]$ for the improved paths changer. This instance shows the real value of the improved paths changer, which can improve the computation time by a significant amount.

Instance 4: Application to a realistic example

In the previous three instances, the improved and original paths changer have been compared. These instances show that the computation time of the paths changing problem can be significantly reduced. To be more specific, the original paths changer randomly looks for new paths, and often changes a route and pair that is not even in conflict. Even if the original paths changer takes a step in the right direction, there is no guarantee that the next attempt will be equally good or better.

Even though the previous three instances show that the improved paths changer significantly improves the search for new paths by a quite significant amount, these three instances obviously do not represent a realistic scenario. Therefore, it is time to look at a more realistic example that could possibly occur in a real industrial application. For that, consider the instance defined in Table 4. In this instance, four jobs are defined, which each need to perform six tasks. Most of the tasks have a specific processing time and time window in which they need to be performed. Furthermore, each job has the same node as start and end location, but these locations are different between the jobs.

Table 4: Jobs information of realistic example

Job	Task	Location	Processing time	Time window
1	**Start**	0	0	-
	1	5	10	[40,40]
	2	54	6	[84,84]
	3	60	0	-
	4	20	5	[175,175]
	5	49	14	[206,206]
	6	29	0	-
	End	0	0	-
2	**Start**	4	0	-
	1	11	20	[70, 70]
	2	45	14	[116,116]
	3	36	0	-
	4	28	34	[206,206]
	5	23	0	[274,274]
	6	16	0	-
	End	4	0	-
3	**Start**	63	0	-
	1	42	20	[70,70]
	2	10	8	[120,120]
	3	62	0	-
	4	9	6	[234,234]
	5	40	0	[278,278]
	6	51	0	-
	End	63	0	-
4	**Start**	59	0	-
	1	52	10	[40,40]
	2	3	6	[84,84]
	3	42	0	-
	4	32	10	[170,170]
	5	19	12	[198,198]
	6	10	0	-
	End	59	0	-

Figure 13 shows the network of nodes and edges corresponding to the problem in Table 4. In this figure, the task locations for each of the jobs have been indicated, as well as the start (and end) location. All of the edges have a length of 4, except for the edges on the outside border, which have a length of 8.

Solution of the improved paths changer

The initial solution to this problem is unsatisfiable. In fact, the capacity verification problem filters out a total of 17 unsatisfiable capacity constraints, divided over 5 unsatisfiable cores. That is, in total there are 17 conflicts in the initial solution as a result of the time windows of the tasks, divided over 5 different combinations of routes and paths. The nodes and edges used by the initial solution can be found in Tables 29 and 30, respectively.

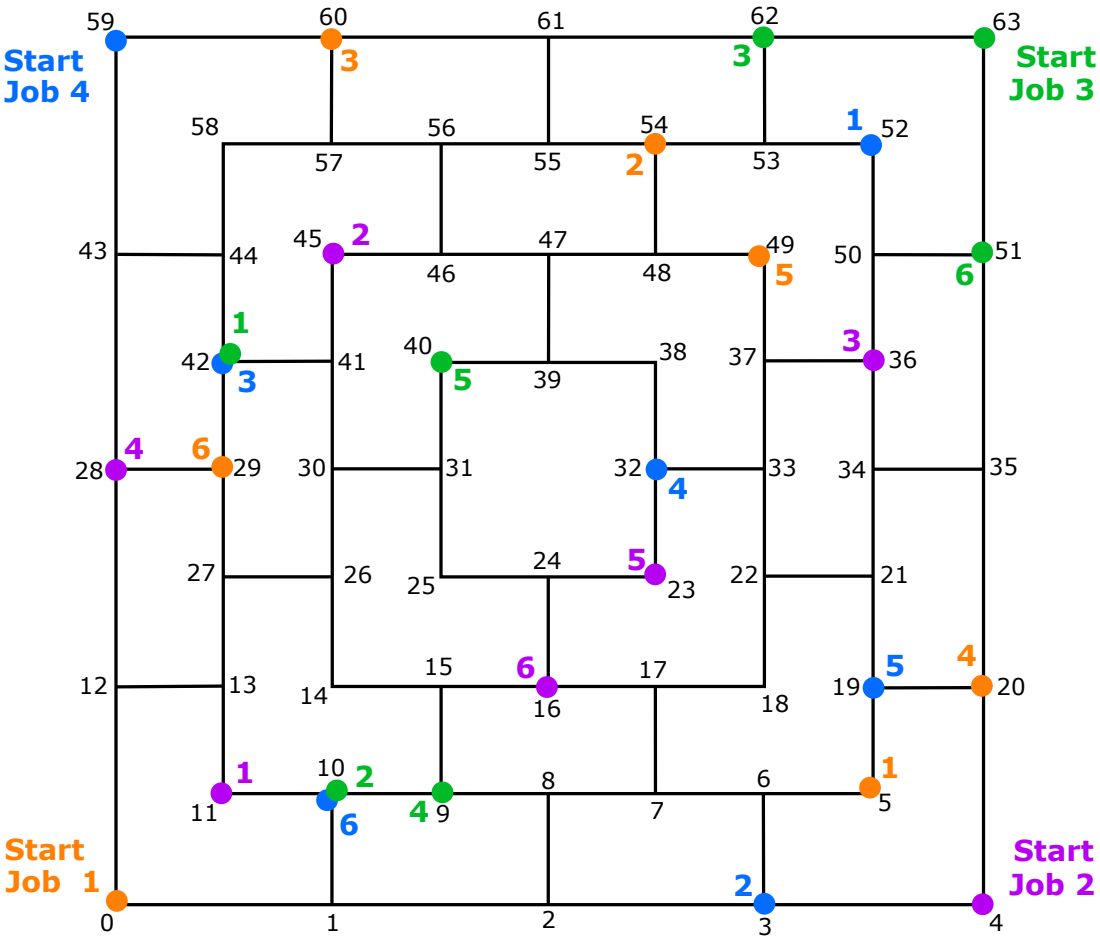


Figure 13: Task locations of realistic example.

The improved paths changing problem finds a feasible solution after only one iteration, which takes 184,6 $[s]$ and has a travelling distance of 824. The nodes and edges used by the improved capacity verification problem can be found in Tables 31 and 32, respectively. For each of the five unsatisfiable cores, let us take a look at what caused the conflict, and how the improved paths changer solved that conflict.

Conflict between route 1 task 1-2 and and route 4 task 1-2

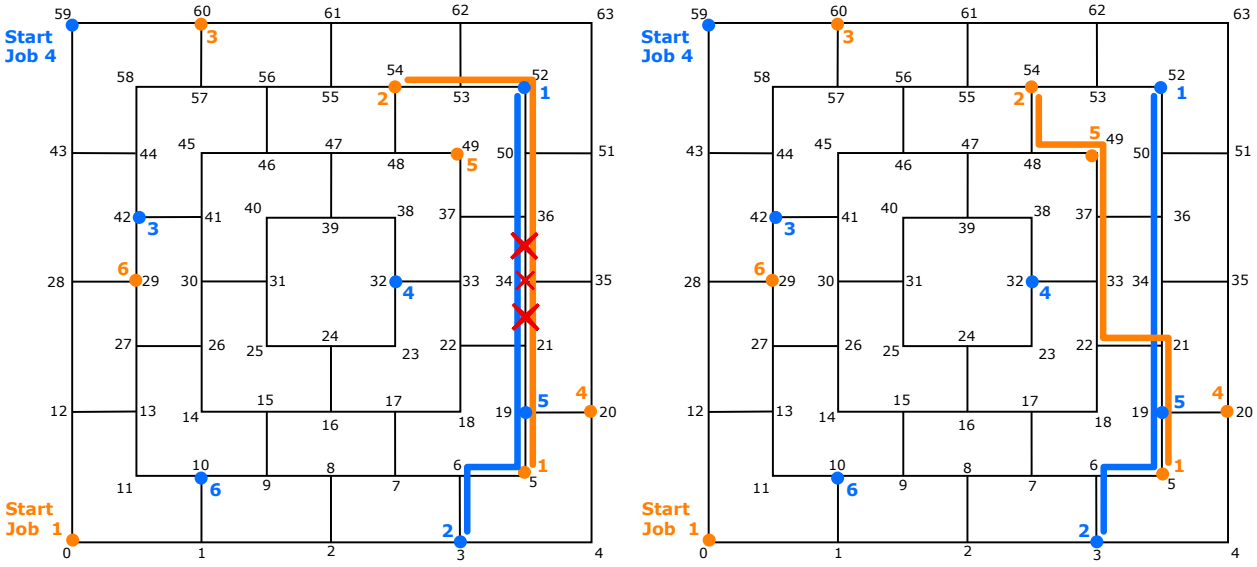
The first conflict that we'll take a look at occurs on the path between route 1 from task 1 to task 2 and route 4 from task 1 to task 2, see Figure 14.

The unsatisfiable core of this conflict, that results because of the time windows of the tasks, consists out of three capacity constraints. These constraints are:

$$\begin{aligned} &\text{Capacity node: R1 P(5,54) N34 - R4 P(52,3) N34} \\ &\text{Capacity edge - opposite: R1 P(5,54) E(21,34) - R4 P(52,3) E(34,21)} \\ &\text{Capacity edge - opposite: R1 P(5,54) E(34,36) - R4 P(52,3) E(36,34)} \end{aligned} \tag{0.0.4}$$

From the unsatisfiable core it can be concluded that there is one node capacity constraint that is violated, and two edge capacity constraints in the opposite direction. In Figure 14a, these conflicts have been indicated by red crosses. Figure 14b, shows the solution of the improved paths changer. As can be seen very clearly,

the conflict has been resolved because route 1 has been rerouted and route 1 - pair (5,54) no longer uses node 34, edge (21,34) and edge (34,36).



(a) Conflict between route 1 task 1-2 and and route 4 task 1-2

(b) Solution to conflict between route 1 task 1-2 and and route 4 task 1-2

Figure 14: Visualization of conflict and solution of route 1 task 1-2 and and route 4 task 1-2.

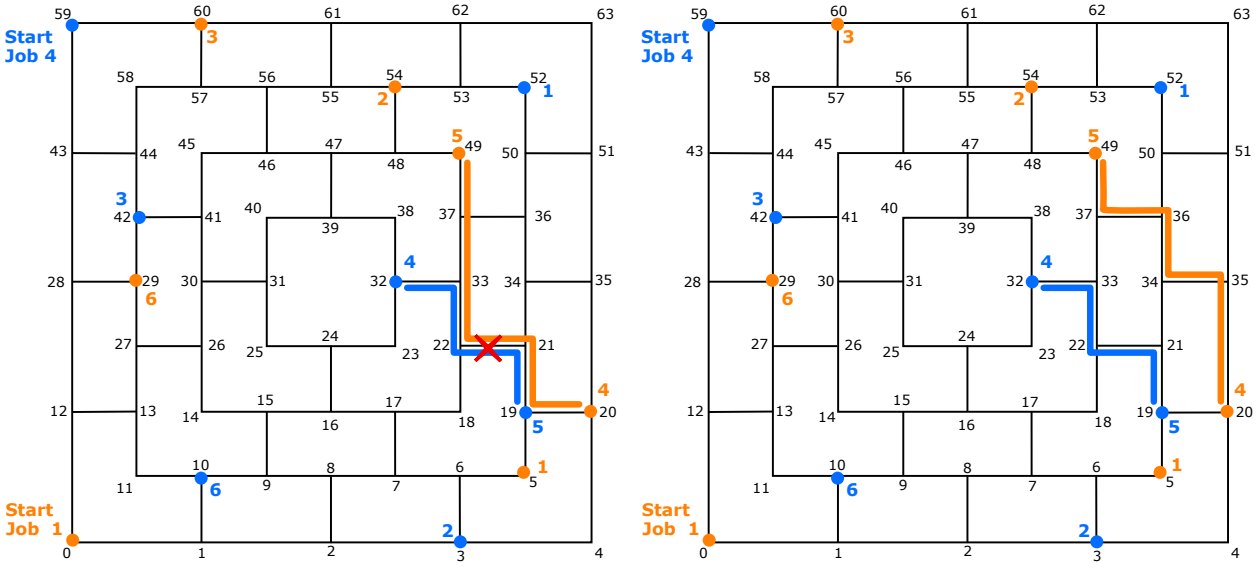
Conflict between route 1 task 4-5 and and route 4 task 4-5

The next conflict that is treated occurs on the path between route 1 from task 4 to task 5 and route 4 from task 4 to task 5, see Figure 15.

The unsatisfiable core of this conflict, which results because of the time windows of the tasks, only consists out of a single capacity constraint, namely:

$$\text{Capacity edge - opposite: R1 P(20,49) E(21,22) - R4 P(32,19) E(22,21)} \tag{0.0.5}$$

So, both route 1 - pair (20,49) and route 4 - pair (32,19) want to use edge (21,22) (or (22,21)), but opposite directions. This conflict is shown in Figure 15a, where this conflict has been indicated with a red cross. Furthermore, a solution to the conflict is shown in Figure 15b, where route 1 - pair (20,49) has been assigned a different path, which resolves the conflict between the two jobs.



(a) Conflict between route 1 task 4-5 and and route 4 task 4-5

(b) Solution to conflict between route 1 task 4-5 and and route 4 task 4-5

Figure 15: Visualization of conflict and solution of route 1 task 4-5 and and route 4 task 4-5.

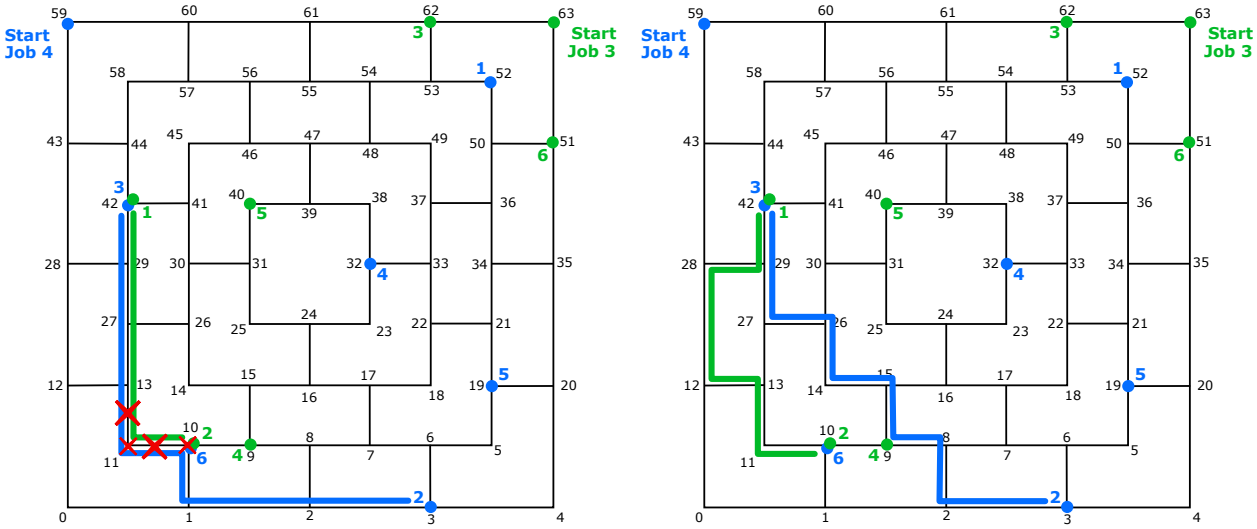
Conflict between route 3 task 1-2 and and route 4 task 2-3

The third conflict that is analyzed occurs on the path between route 3 from task 1 to task 2 and route 4 from task 2 to task 3, see Figure 16.

The unsatisfiable core of this conflict has four unsatisfiable capacity constraints, resulting from the time windows of the tasks, and looks as follows:

$$\begin{aligned}&\text{Capacity node: R3 P(42,10) N10 - R4 P(3,42) N10}\\&\text{Capacity node: R3 P(42,10) N11 - R4 P(3,42) N11}\\&\text{Capacity edge - opposite: R3 P(42,10) E(11,10) - R4 P(3,42) E(10,11)}\\&\text{Capacity edge - opposite: R3 P(42,10) E(13,11) - R4 P(3,42) E(11,13)}\end{aligned} \tag{0.0.6}$$

In this conflict, two node capacity constraints are violated, as well as two edge capacity constraints in opposite directions. The conflict and solution of the improved paths changer are shown in figures 16a and 16b, respectively. The conflicts have been indicated with red crosses. Node 10 is a job location for job 3, so route 3 specifically has to use this node. This does not necessarily mean that route 4 has to be the one that is rerouted, as long as job 3 can still visit node 10. However, in this scenario rerouting route 4 leads to a shorter distance traveled than rerouting route 3. Nevertheless, route 3 also takes a slight detour (via node 12 and 28). The reason for this is another conflict that happens, namely between route 2 task 1-2 route 3 task 1-2, which is investigated next.



(a) Conflict between route 3 task 1-2 and and route 4 task 2-3

(b) Solution to conflict between route 3 task 1-2 and and route 4 task 2-3

Figure 16: Visualization of conflict between route 3 task 1-2 and and route 4 task 2-3.

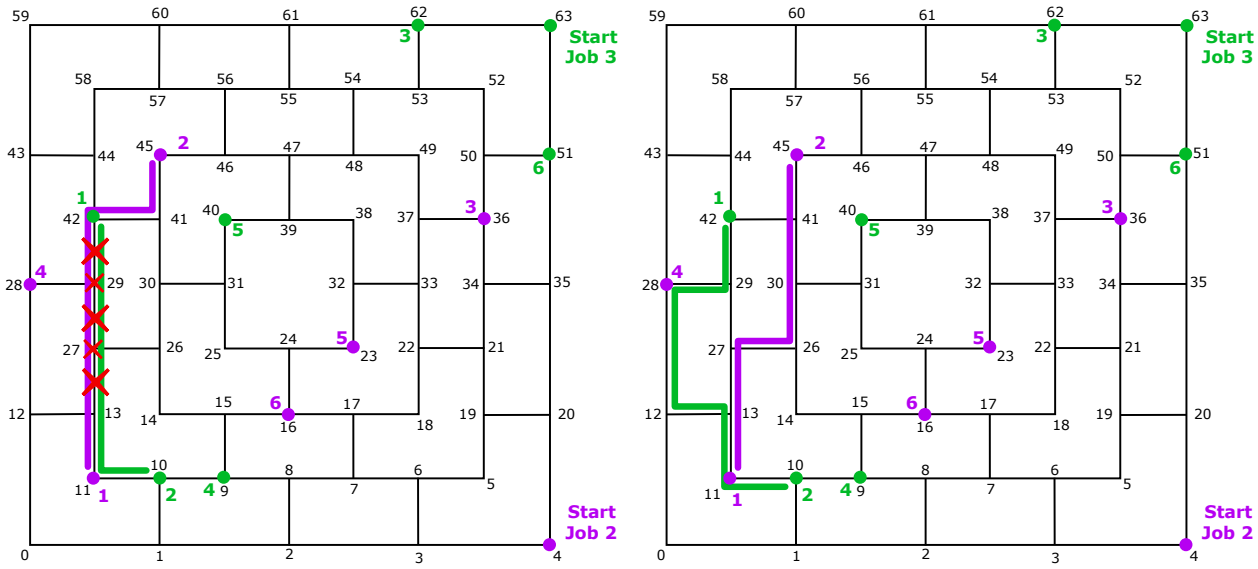
Conflict between route 2 task 1-2 and and route 3 task 1-2

The following conflict that we'll treat occurs on the path between route 2 from task 1 to task 2 and route 3 from task 1 to task 2, see Figure 17,

As a result of the time windows of the tasks, there are five unsatisfiable capacity constraints in the unsatisfiable core, namely :

$$\begin{aligned}
&\text{Capacity node: R2 P(11,45) N27 - R3 P(42,10) N27}\\
&\text{Capacity node: R2 P(11,45) N27 - R3 P(42,10) N29}\\
&\text{Capacity edge - opposite: R2 P(11,45) E(13,27) - R3 P(42,10) E(27,13)} \qquad (0.0.7)\\
&\text{Capacity edge - opposite: R2 P(11,45) E(27,29) - R3 P(42,10) E(29,27)}\\
&\text{Capacity edge - opposite: R2 P(11,45) E(29,42) - R3 P(42,10) E(42,29)}
\end{aligned}$$

As can be seen, two node capacity constraints and three edge capacity constraints in opposite directions are violated. These five conflicts are shown in Figure 17a, and are indicated with red crosses. A solution to the conflicts is shown in Figure 17b. As can be seen, route 3 does not use node 27, edge (27,13) and edge (29,27). Additionally, route 2 does not use node 29, edge (29,42) and edge (27,29). Changing these two paths, as well as the path of route 4 task 2-3, ensures that all three paths are feasible. Now, they 'work together' such that each of the routes can use part of the nodes and edges that were in conflict in the initial solution.



(a) Conflict between route 2 task 1-2 and and route 3 task 1-2

(b) Solution to conflict between route 2 task 1-2 and and route 3 task 1-2

Figure 17: Visualization of conflict between route 2 task 1-2 and and route 3 task 1-2.

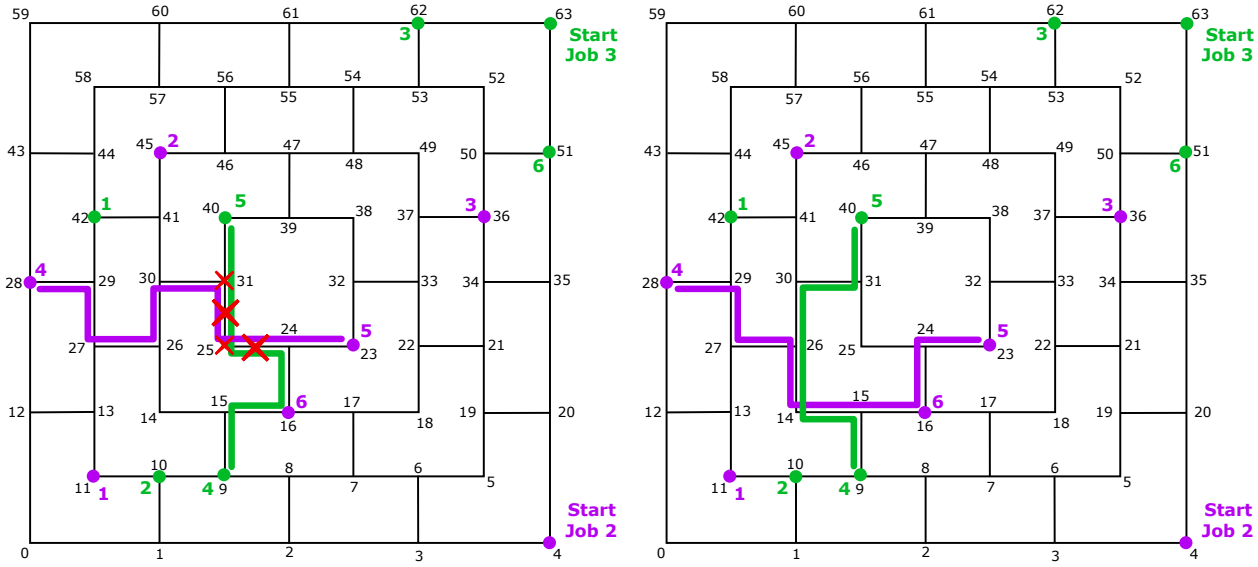
Conflict between route 2 task 4-5 and and route 3 task 4-5

The last conflict occurs on the path between route 2 from task 5 to task 5 and route 3 from task 4 to task 5, see Figure 18.

This conflict has four unsatisfiable capacity constraints as a result of the time windows of the tasks, which are:

$$\begin{aligned}&\text{Capacity node: R2 P(28,23) N25 - R3 P(9,40) N25}\\&\text{Capacity node: R2 P(28,23) N31 - R3 P(9,40) N31}\\&\text{Capacity edge - opposite: R2 P(28,23) E(25,24) - R3 P(9,40) E(24,25)}\\&\text{Capacity edge - opposite: R2 P(28,23) E(31,25) - R3 P(9,40) E(25,31)}\end{aligned} \tag{0.0.8}$$

In this unsatisfiable core, two node capacity constraints and two edge capacity constraints in opposite directions can be observed. These conflicts are shown in Figure 18a indicated by red crosses. Furthermore, Figure 18a shows a solution of the improved paths changer. As can be seen, both routes have been redirected. The constraints for the *PathChanger* defined based on the UNSAT core specify that at least one of the two conflicting combinations of routes and pairs cannot use the conflicting nodes and edges. However, in this situation, the paths of both routes have been changed. If one measures the lengths, it can be seen that route 2 task 4-5 has a length of 32 in both cases, and route 3 task 4-5 has a length of 24 in both cases. So, the total distance traveled has remained the same. What can be noted here is that the paths changer simply searches for the solution with the shortest distance traveled, as long as the above constraint is satisfied. This can also mean that both conflicting combinations of routes and pairs are changed. Out of all situations where the total distance of route 2 task 4-5 and route 3 task 4-5 is 56, an arbitrary one is chosen which satisfies all constraints.



(a) Conflict between route 2 task 4-5 and and route 3 task 4-5

(b) Solution to conflict between route 2 task 4-5 and and route 3 task 4-5

Figure 18: Visualization of conflict between route 2 task 1-2 and and route 3 task 4-5.

Solution of the original paths changer

Instead of using the improved paths changer, it is also interesting how the original paths changer performs for the realistic example. As it turns out, the original paths changer requires 15 iterations in order to find a feasible solution, as opposed to 1 iteration for the improved paths changer. The nodes and edges used by the fifteenth iteration of the original paths changer can be found in Tables 33 and 34, respectively. In comparison to instances 2 and 3, this number of iterations is actually not that bad. However, these 15 iterations take 643.4 $[s]$ to compute. In comparison to the computation time of the improved paths changer, which took 184,6 $[s]$ to find a feasible solution, this is obviously much worse. From this, and the previous three instances, it is safe to conclude that the improved paths changer really has the potential to add a lot of value to the *ComSat* algorithm.

Instance 5: Nuancing the results of the original paths changer

Truth to be told, one crucial scenario was found in which the original paths changer has a better performance than the improved paths changer. Consider the instance shown in Figure 19. This instance is for a large part the same as the realistic instance of the previous section, shown in Figure 13, but the task locations of job 3 task 1, and job 3 task 2 have been changed from location 42 to 58, and from location 10 to 26, respectively. Furthermore, the time window of job 3 task 2 has been changed from $[120, 120]$ to $[112, 112]$. In this instance, the improved paths changer finds a solution after one iteration, with a distance of 792, in 128.4 $[s]$. However, the original paths changer also finds a solution after one iteration, with a distance of 792, but only requires 21.2 $[s]$. So, even though the instance is almost entirely the same, the original paths changer is now more than 30 times faster than for the previous instance. Moreover, in this instance it is more than 6 times faster than the improved paths changer.

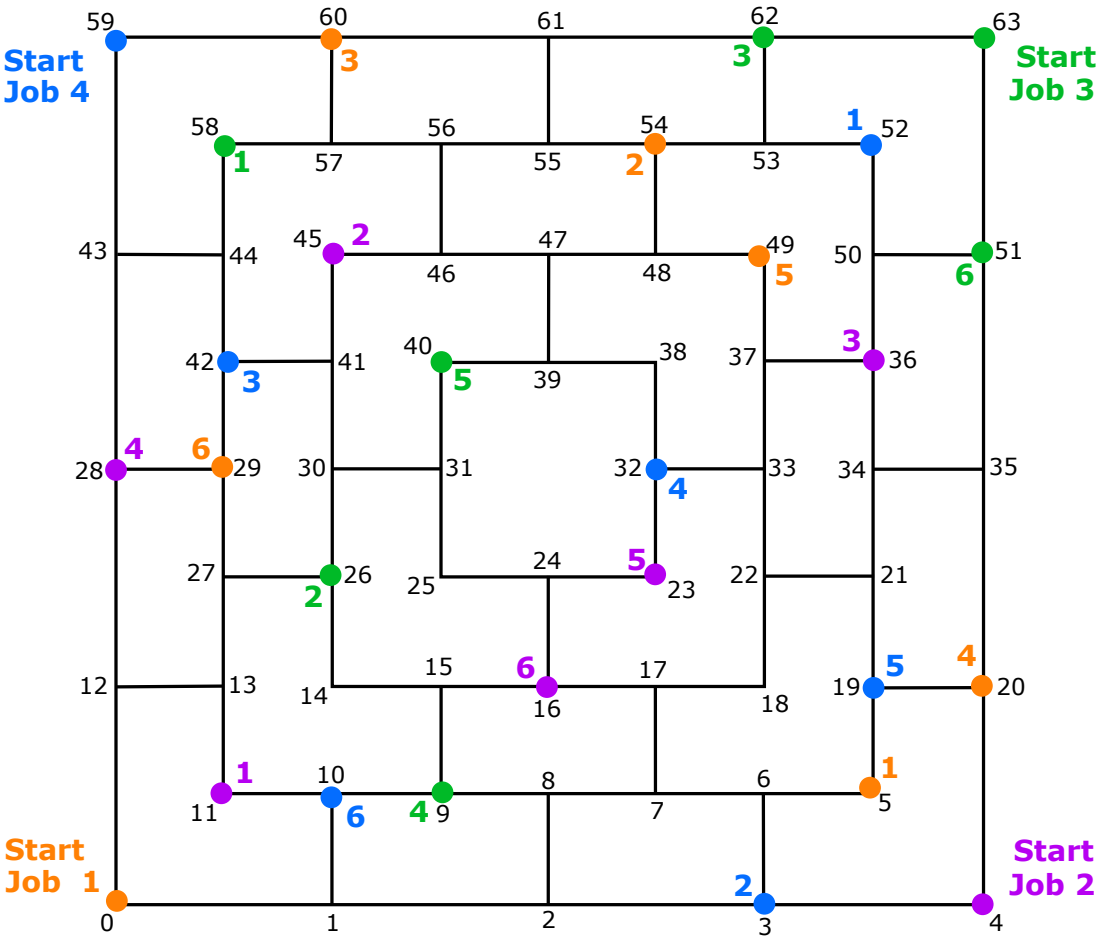


Figure 19: Task locations of nuanced realistic instance.

As was illustrated by instances 1, 2 and 3, the original paths changer just selects the set of paths with the smallest distance traveled, or randomly selects one of them if there are multiple sets of paths with the same distance. However, many of these sets of paths with the smallest distance traveled can be infeasible, so the original paths changing algorithm keeps trying infeasible solutions. Moreover, this algorithm has no idea when it has just solved part of the conflicts, and it is not able to use this information for the next attempt. In conclusion, the number of iterations (and thus also computation time) required by the original paths changer is more a matter of probability than a deliberate decision. In this nuanced realistic instance, the original paths changer happens to select a set of paths within the red circle largely because of coincidence. In some cases, this might lead to a lower computation time than the improved paths changer because the improved paths changer always has to calculate the set of satisfiable paths (or rather, the set of paths that has not yet been evaluated as unsatisfiable), but this is rarely ever the case.

Bibliography

[1] K. Azadeh, M. deKoster, D. Roy. "Robotized warehouse systems: Developments and research opportunities," *ERIM Report Series Research in Management*, no. ERS-2017-009-LIS, 2017.

[2] N. Bjørner, L. de Moura, L. Nachmanson, C. Wintersteiger. "Programming Z3". `http://theory.stanford.edu/~nikolaj/programmingz3.html`.

[3] Computerphile. (2017, January 4). "Dijkstra's Algorithm - Computerphile" [Video]. YouTube. `https://www.youtube.com/watch?v=GazC3A4OQTE`

[4] Fisher, H. and Thompson, G. L. (1963). "Probabilistic learning combinations of local job shop scheduling rules.", Prentice-Hall, Englewood Cliffs, pp. 225-251.

[5] Diarmuid Grimes, Emmanuel Hebrard. "Job Shop Scheduling with Setup Times and Maximal Time-Lags: A Simple Constraint Programming Approach." *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems*, 2010, pp. 147–161., `https://doi.org/10.1007/978-3-642-13520-0_19`.

[6] Diarmuid Grimes, Emmanuel Hebrard. "Models and Strategies for Variants of the Job Shop Scheduling Problem." *Principles and Practice of Constraint Programming – CP 2011*, Sept. 2011, pp. 356–372., `https://doi.org/10.1007/978-3-642-23786-7_28`.

[7] I. Lynce, J. Marques-Silva. (2010). "On Computing Minimum Unsatisfiable Cores." *IST/INESC-ID, Technical University of Lisbon, Portugal*

[8] Amit Patel. "Introduction to A*". `http://theory.stanford.edu/~amitp/GameProgramming/AStarComparison.html`.

[9] S.F. Roselli, K. Åkesson, K. Bengtsson. "SMT Solvers for Job-Shop Scheduling Problems: Models Comparison and Performance Evaluation." *2018 IEEE 14th International Conference on Automation Science and Engineering (CASE)*, August 2018, `https://doi.org/10.1109/coase.2018.8560344`.

[10] S.F. Roselli, M. Fabian, K. Åkesson. "An SMT Based Compositional Algorithm to Solve a Conflict-Free Electric Vehicle Routing Problem." *2021 IEEE 17th International Conference on Automation Science and Engineering (CASE)*, August, 2021, `https://doi.org/10.1109/case49439.2021.9551521`.

[11] Dennis Yurichev. "SAT/SMT by Example." November, 2021, `https://sat-smt.codes/SAT_SMT_by_example.pdf`.

Appendix

This appendix presents the results of all examples of presented above. For each example, the used nodes and edges of the initial solution, the improved paths changer, the first attempt of the original paths changer, and the last attempt of the original paths changer are indicated in separate tables.

Example 1

Initial solution

Table 5: Nodes initial solution: Unsatisfiable

Job	Pair	Nodes
1	(0,1)	0, 1
	(1,0)	1, 0
2	(1,0)	1, 0
	(0,1)	0, 1

Table 6: Edges initial solution: Unsatisfiable

Job	Pair	Edges
1	(0,1)	(0,1)
	(1,0)	(1,0)
2	(1,0)	(1,0)
	(0,1)	(0,1)

Improved paths changer

Table 7: Nodes improved paths changer - Attempt 1: Satisfiable

Job	Pair	Nodes
1	(0,1)	0, 1
	(1,0)	1, 0
2	(1,0)	1, 2, 0
	(0,1)	0, 1

Table 8: Edges improved paths changer - Attempt 1: Satisfiable

Job	Pair	Edges
1	(0,1)	(0,1)
	(1,0)	(1,0)
2	(1,0)	(1,2), (2,0)
	(0,1)	(0,1)

Original paths changer

Table 9: Nodes original paths changer - Attempt 1: Unsatisfiable

Job	Pair	Nodes
1	(0,1)	0, 1
	(1,0)	1, 0
2	(1,0)	1, 0
	(0,1)	0, 2, 1

Table 10: Edges original paths changer - Attempt 1: Unsatisfiable

Job	Pair	Edges
1	(0,1)	(0,1)
	(1,0)	(1,0)
2	(1,0)	(1,0)
	(0,1)	(0,2), (2,1)

Table 11: Nodes original paths changer - Attempt 2: Satisfiable

Job	Pair	Nodes
1	(0,1)	0, 2, 1
	(1,0)	1, 0
2	(1,0)	1, 0
	(0,1)	0, 1

Table 12: Edges original paths changer - Attempt 2: Satisfiable

Job	Pair	Edges
1	(0,1)	(0,2), (2,1)
	(1,0)	(1,0)
2	(1,0)	(1,0)
	(0,1)	(0,1)

Example 2

Initial solution

Table 13: Nodes initial solution: Unsatisfiable

Job	Pair	Nodes
1	(0,3)	0, 1, 2, 3
	(3,0)	3, 2, 1, 0
2	(0,3)	0, 1, 2, 3
	(3,0)	3, 2, 1, 0
3	(3,0)	3, 2, 1, 0
	(0,3)	0, 1, 2, 3

Table 14: Edges initial solution: Unsatisfiable

Job	Pair	Nodes
1	(0,3)	(0,1), (1,2), (2,3)
	(3,0)	(3,2), (2,1), (1,0)
2	(0,3)	(0,1), (1,2), (2,3)
	(3,0)	(3,2), (2,1), (1,0)
3	(3,0)	(3,2), (2,1), (1,0)
	(0,3)	(0,1), (1,2), (2,3)

Improved paths changer

Table 15: Nodes improved paths changer - Attempt 1: Satisfiable

Job	Pair	Nodes
1	(0,3)	0, 1, 2, 3
	(3,0)	3, 2, 1, 0
2	(0,3)	0, 1, 2, 3
	(3,0)	3, 2, 1, 0
3	(3,0)	3, 2, 5, 6, 1, 0
	(0,3)	0, 1, 2, 3

Table 16: Edges improved paths changer - Attempt 1: Satisfiable

Job	Pair	Edges
1	(0,3)	(0,1), (1,2), (2,3)
	(3,0)	(3,2), (2,1), (1,0)
2	(0,3)	(0,1), (1,2), (2,3)
	(3,0)	(3,2), (2,1), (1,0)
3	(3,0)	(3,2), (2,5), (5,6), (6,1), (1,0)
	(0,3)	(0,1), (1,2), (2,3)

Original paths changer

Table 17: Nodes original paths changer - Attempt 1: Unsatisfiable

Job	Pair	Nodes
1	(0,3)	0, 1, 2, 3
	(3,0)	3, 4, 5, 6, 7, 0
2	(0,3)	0, 1, 2, 3
	(3,0)	3, 2, 1, 0
3	(3,0)	3, 2, 1, 0
	(0,3)	0, 1, 2, 3

Table 18: Edges original paths changer - Attempt 1: Unsatisfiable

Job	Pair	Edges
1	(0,3)	(0,1), (1,2), (2,3)
	(3,0)	(3,4), (4,5), (5,6), (6,7), (7,0)
2	(0,3)	(0,1), (1,2), (2,3)
	(3,0)	(3,2), (2,1), (1,0)
3	(3,0)	(3,2), (2,1), (1,0)
	(0,3)	(0,1), (1,2), (2,3)

Table 19: Nodes original paths changer - Attempt 24: Satisfiable

Job	Pair	Nodes
1	(0,3)	0, 1, 2, 3
	(3,0)	3, 2, 1, 0
2	(0,3)	0, 1, 2, 3
	(3,0)	3, 2, 1, 0
3	(3,0)	3, 2, 5, 6, 7, 0
	(0,3)	0, 1, 2, 3

Table 20: Edges original paths changer - Attempt 24: Satisfiable

Job	Pair	Edges
1	(0,3)	(0,1), (1,2), (2,3)
	(3,0)	(3,2), (2,1), (1,0)
2	(0,3)	(0,1), (1,2), (2,3)
	(3,0)	(3,2), (2,1), (1,0)
3	(3,0)	(3,2), (2,5), (5,6), (6,7), (7,0)
	(0,3)	(0,1), (1,2), (2,3)

Example 3

Initial solution

Table 21: Nodes initial solution: Unsatisfiable

Job	Pair	Nodes
1	(0,2)	0, 4, 2
	(2,0)	2, 4, 0
2	(1,3)	1, 4, 3
	(3,1)	3, 4, 1
3	(2,0)	2, 4, 0
	(0,2)	0, 4, 2
4	(3,1)	3, 4, 1
	(1,3)	1, 4, 3

Table 22: Edges initial solution: Unsatisfiable

Job	Pair	Edges
1	(0,2)	(0,4), (4,2)
	(2,0)	(2,4), (4,0)
2	(1,3)	(1,4), (4,3)
	(3,1)	(3,4), (4,1)
3	(2,0)	(2,4), (4,0)
	(0,2)	(0,4), (4,2)
4	(3,1)	(3,4), (4,1)
	(1,3)	(1,4), (4,3)

Improved paths changer

Table 23: Nodes improved paths changer - Attempt 1: Satisfiable

Job	Pair	Nodes
1	(0,2)	0, 4, 2
	(2,0)	2, 4, 0
2	(1,3)	1, 4, 3
	(3,1)	3, 4, 1
3	(2,0)	2, 3, 0
	(0,2)	0, 4, 2
4	(3,1)	3, 0, 1
	(1,3)	1, 4, 3

Table 24: Edges improved paths changer - Attempt 1: Satisfiable

Job	Pair	Edges
1	(0,2)	(0,4), (4,2)
	(2,0)	(2,4), (4,0)
2	(1,3)	(1,4), (4,3)
	(3,1)	(3,4), (4,1)
3	(2,0)	(2,3), (3,0)
	(0,2)	(0,4), (4,2)
4	(3,1)	(3,0), (0,1)
	(1,3)	(1,4), (4,3)

Original paths changer

Table 25: Nodes original paths changer - Attempt 1: Unsatisfiable

Job	Pair	Nodes
1	(0,2)	0, 4, 2
	(2,0)	2, 4, 0
2	(1,3)	1, 4, 3
	(3,1)	3, 0, 1
3	(2,0)	2, 4, 0
	(0,2)	0, 4, 2
4	(3,1)	3, 4, 1
	(1,3)	1, 4, 3

Table 26: Edges original paths changer - Attempt 1: Unsatisfiable

Job	Pair	Edges
1	(0,2)	(0,4), (4,2)
	(2,0)	(2,4), (4,0)
2	(1,3)	(1,4), (4,3)
	(3,1)	(3,0), (0,1)
3	(2,0)	(2,4), (4,0)
	(0,2)	(0,4), (4,2)
4	(3,1)	(3,4), (4,1)
	(1,3)	(1,4), (4,3)

Table 27: Nodes original paths changer - Attempt 54: Satisfiable

Job	Pair	Nodes
1	(0,2)	0, 4, 2
	(2,0)	2, 4, 0
2	(1,3)	1, 2, 3
	(3,1)	3, 4, 1
3	(2,0)	2, 3, 0
	(0,2)	0, 4, 2
4	(3,1)	3, 4, 1
	(1,3)	1, 4, 3

Table 28: Edges original paths changer - Attempt 54: Satisfiable

Job	Pair	Edges
1	(0,2)	(0,4), (4,2)
	(2,0)	(2,4), (4,0)
2	(1,3)	(1,2), (2,3)
	(3,1)	(3,4), (4,1)
3	(2,0)	(2,3), (3,0)
	(0,2)	(0,4), (4,2)
4	(3,1)	(3,4), (4,1)
	(1,3)	(1,4), (4,3)

Realistic example

Initial solution

Table 29: Nodes initial solution: Unsatisfiable

Job	Pair	Nodes
1	(0,5)	0, 1, 2, 3, 6, 5
	(5,54)	5, 19, 21, 34, 36, 50, 52, 53, 54
	(54,60)	54, 55, 61, 60
	(60,20)	60, 61, 62, 63, 51, 35, 20
	(20,49)	20, 19, 21, 22, 33, 37, 49
	(49,29)	49, 48, 47, 46, 45, 41, 42, 29
	(29,0)	29, 28, 12, 0
2	(4,11)	4, 3, 2, 1, 10, 11
	(11,45)	11, 13, 27, 29, 42, 41, 45
	(45,36)	45, 46, 47, 48, 49, 37, 36
	(36,28)	36, 37, 49, 48, 47, 46, 45, 41, 42, 29, 28
	(28,23)	28, 29, 27, 26, 30, 31, 25, 24, 23
	(23,16)	23, 24, 16
	(16,4)	16, 17, 7, 6, 3, 4
3	(63,42)	63, 62, 61, 55, 56, 57, 58, 44, 42
	(42,10)	42, 29, 27, 13, 11, 10
	(10,62)	10, 9, 8, 7, 6, 5, 19, 21, 34, 36, 50, 52, 53, 62
	(62,9)	62, 61, 55, 56, 46, 45, 41, 30, 26, 14, 15, 9
	(9,40)	9, 15, 16, 24, 25, 31, 40
	(40,51)	40, 39, 47, 48, 54, 53, 52, 50, 51
	(51,63)	51, 63
4	(59,52)	59, 60, 61, 62, 53, 52
	(52,3)	52, 50, 36, 34, 21, 19, 5, 6, 3
	(3,42)	3, 2, 1, 10, 11, 13, 27, 29, 42
	(42,32)	42, 41, 30, 31, 25, 24, 23, 32
	(32,19)	32, 33, 22, 21, 19
	(19,10)	19, 5, 6, 7, 8, 9, 10
	(10,59)	10, 11, 13, 12, 28, 43, 59

Table 30: Edges initial solution: Unsatisfiable

Job	Pair	Edges
1	(0,5)	(0,1), (1,2), (2,3), (3,6), (6,5)
	(5,54)	(5,19), (19,21), (21,34), (34,36), (36,50), (50,52), (52,53), (53,54)
	(54,60)	(54,55), (55,61), (61,60)
	(60,20)	(60,61), (61,62), (62,63), (63,51), (51,35), (35,20)
	(20,49)	(20,19), (19,21), (21,22), (22,33), (33,37), (37,49)
	(49,29)	(49,48), (48,47), (47,46), (46,45), (45,41), (41,42), (42,29)
	(29,0)	(29,28), (28,12), (12,0)
2	(4,11)	(4,3), (3,2), (2,1), (1,10), (10,11)
	(11,45)	(11,13), (13,27), (27,29), (29,42), (42,41), (41,45)
	(45,36)	(45,46), (46,47), (47,48), (48,49), (49,37), (37,36)
	(36,28)	(36,37), (37,49), (49,48), (48,47), (47,46), (46,45), (45,41), (41,42), (42,29), (29,28)
	(28,23)	(28,29), (29,27), (27,26), (26,30), (30,31), (31,25), (25,24), (24,23)
	(23,16)	(23,24), (24,16)
	(16,4)	(16,17), (17,7), (7,6), (6,3), (3,4)
3	(63,42)	(63,62), (62,61), (61,55), (55,56), (56,57), (57,58), (58,44), (44,42)
	(42,10)	(42,29), (29,27), (27,13), (13,11), (11,10)
	(10,62)	(10,9), (9,8), (8,7), (7,6), (6,5), (5,19), (19,21), (21,34), (34,36), (36,50), (50,52), (52,53), (53,62)
	(62,9)	(62,61), (61,55), (55,56), (56,46), (46,45), (45,41), (41,30), (30,26), (26,14), (14,15), (15,9)
	(9,40)	(9,15), (15,16), (16,24), (24,25), (25,31), (31,40)
	(40,51)	(40,39), (39,47), (47,48), (48,54), (54,53), (53,52), (52,50), (50,51)
	(51,63)	(51,63)
4	(59,52)	(59,60), (60,61), (61,62), (62,53), (53,52)
	(52,3)	(52,50), (50,36), (36,34), (34,21), (21,19), (19,5), (5,6), (6,3)
	(3,42)	(3,2), (2,1), (1,10), (10,11), (11,13), (13,27), (27,29), (29,42)
	(42,32)	(42,41), (41,30), (30,31), (31,25), (25,24), (24,23), (23,32)
	(32,19)	(32,33), (33,22), (22,21), (21,19)
	(19,10)	(19,5), (5,6), (6,7), (7,8), (8,9), (9,10)
	(10,59)	(10 11), (11,13), (13,12), (12,28), (28,43), (43,59)

Improved paths changer

Table 31: Nodes improved paths changer - Attempt 1: Satisfiable

Job	Pair	Nodes
1	(0,5)	0, 1, 2, 8, 7, 6, 5
	(5,54)	5, 19, 21, 22, 33, 37, 49, 48, 54
	(54,60)	54, 55, 61, 60
	(60,20)	60, 57, 56, 46, 47, 48, 49, 37, 33, 22, 21, 19, 20
	(20,49)	20, 35, 34, 36, 37, 49
	(49,29)	49, 48, 47, 46, 45, 41, 42, 29
	(29,0)	29, 27, 13, 12, 0
2	(4,11)	4, 3, 2, 8, 9, 10, 11
	(11,45)	11, 13, 27, 26, 30, 41, 45
	(45,36)	45, 46, 47, 48, 49, 37, 36
	(36,28)	36, 37, 49, 48, 47, 46, 45, 41, 42, 29, 28
	(28,23)	28, 29, 27, 26, 14, 15, 16, 24, 23
	(23,16)	23, 24, 16
	(16,4)	16, 17, 7, 6, 3, 4
3	(63,42)	63, 62, 61, 60, 57, 58, 44, 42
	(42,10)	42, 29, 28, 12, 13, 11, 10
	(10,62)	10, 11, 13, 27, 29, 42, 44, 58, 57, 60, 61, 62
	(62,9)	62, 53, 54, 48, 47, 39, 40, 31, 30, 26, 14, 15, 9
	(9,40)	9, 15, 14, 26, 30, 31, 40
	(40,51)	40, 39, 47, 48, 49, 37, 36, 50, 51
	(51,63)	51, 63
4	(59,52)	59, 60, 57, 56, 55, 54, 53, 52
	(52,3)	52, 50, 36, 34, 21, 19, 5, 6, 3
	(3,42)	3, 2, 8, 9, 15, 14, 26, 27, 29, 42
	(42,32)	42, 41, 45, 46, 47, 39, 38, 32
	(32,19)	32, 33, 22, 21, 19
	(19,10)	19, 5, 6, 7, 8, 9, 10
	(10,59)	10, 11, 13, 12, 28, 43, 59

Table 32: Edges improved paths changer - Attempt 1: Satisfiable

Job	Pair	Edges
1	(0,5)	(0,1), (1,2), (2,8), (8,7), (7,6), (6,5)
	(5,54)	(5,19), (19,21), (21,22), (22,33), (33,37), (37,49), (49,48), (48,54)
	(54,60)	(54,55), (55,61), (61,60)
	(60,20)	(60,57), (57,56), (56,46), (46,47), (47,48), (48,49), (49,37), (37,33), (33,22), (22,21), (21,19), (19,20)
	(20,49)	(20,35), (35,34), (34,36), (36,37), (37,49)
	(49,29)	(49,48), (48,47), (47,46), (46,45), (45,41), (41,42), (42,29)
	(29,0)	(29,27), (27,13), (13,12), (12,0)
2	(4,11)	(4,3), (3,2), (2,8), (8,9), (9,10), (10,11)
	(11,45)	(11,13), (13,27), (27,26), (26,30), (30,41), (41,45)
	(45,36)	(45,46), (46,47), (47,48), (48,49), (49,37), (37,36)
	(36,28)	(36,37), (37,49), (49,48), (48,47), (47,46), (46,45), (45,41), (41,42), (42,29), (29,28)
	(28,23)	(28,29), (29,27), (27,26), (26,14), (14,15), (15,16), (16,24), (24,23)
	(23,16)	(23,24), (24,16)
	(16,4)	(16,17), (17,7), (7,6), (6,3), (3,4)
3	(63,42)	(63,62), (62,61), (61,60), (60,57), (57,58), (58,44), (44,42)
	(42,10)	(42,29), (29,28), (28,12), (12,13), (13,11), (11,10)
	(10,62)	(10,11), (11,13), (13,27), (27,29), (29,42), (42,44), (44,58), (58,57), (57,60), (60,61), (61,62)
	(62,9)	(62,53), (53,54), (54,48), (48,47), (47,39), (39,40), (40,31), (31,30), (30,26), (26,14), (14,15), (15, 9)
	(9,40)	(9,15), (15,14), (14,26), (26,30), (30,31), (31,40)
	(40,51)	(40,39), (39,47), (47,48), (48,49), (49,37), (37,36), (36,50), (50,51)
	(51,63)	(51,63)
4	(59,52)	(59,60), (60,57), (57,56), (56,55), (55,54), (54,53), (53,52)
	(52,3)	(52,50), (50,36), (36,34), (34,21), (21,19), (19,5), (5,6), (6,3)
	(3,42)	(3,2), (2,8), (8,9), (9,15), (15,14), (14,26), (26,27), (27,29), (29,42)
	(42,32)	(42,41), (41,45), (45,46), (46,47), (47,39), (39,38), (38,32)
	(32,19)	(32,33), (33,22), (22,21), (21,19)
	(19,10)	(19,5), (5,6), (6,7), (7,8), (8,9), (9,10)
	(10,59)	(10,11), (11,13), (13,12), (12,28), (28,43), (43,59)

Original paths changer

Table 33: Nodes original paths changer - Attempt 15: Satisfiable

Job	Pair	Nodes
1	(0,5)	0, 1, 2, 8, 7, 6, 5
	(5,54)	5, 19, 21, 22, 33, 37, 49, 48, 54
	(54,60)	54, 55, 56, 57, 60
	(60,20)	60, 61, 55, 54, 53, 52, 50, 51, 35, 20
	(20,49)	20, 19, 21, 34, 36, 37, 49
	(49,29)	49, 48, 47, 46, 45, 41, 42, 29
	(29,0)	29, 28, 12, 0
2	(4,11)	4, 3, 2, 8, 9, 10, 11
	(11,45)	11, 13, 27, 26, 30, 41, 45
	(45,36)	45, 46, 47, 48, 49, 37, 36
	(36,28)	36, 37, 49, 48, 47, 46, 45, 41, 42, 29, 28
	(28,23)	28, 29, 27, 26, 14, 15, 16, 24, 23
	(23,16)	23, 24, 16
	(16,4)	16, 17, 7, 6, 3, 4
3	(63,42)	63, 62, 53, 54, 55, 56, 57, 58, 44, 42
	(42,10)	42, 29, 27, 13, 11, 10
	(10,62)	10, 11, 13, 27, 29, 42, 44, 58, 57, 56, 55, 61, 62
	(62,9)	62, 53, 54, 55, 56, 46, 45, 41, 30, 26, 14, 15, 9
	(9,40)	9, 15, 16, 24, 25, 31, 40
	(40,51)	40, 39, 47, 48, 49, 37, 36, 50, 51
	(51,63)	51, 63
4	(59,52)	59, 60, 61, 55, 54, 53, 52
	(52,3)	52, 50, 36, 34, 21, 19, 5, 6, 3
	(3,42)	3, 6, 7, 17, 16, 15, 14, 26, 27, 29, 42
	(42,32)	42, 41, 30, 31, 25, 24, 23, 32
	(32,19)	32, 33, 22, 21, 19
	(19,10)	19, 5, 6, 7, 8, 9, 10
	(10,59)	10, 11, 13, 12, 28, 43, 59

Table 34: Edges original paths changer - Attempt 15: Satisfiable

Job	Pair	Edges
1	(0,5)	(0,1), (1,2), (2,8), (8,7), (7,6), (6,5)
	(5,54)	(5,19), (19,21), (21,22), (22,33), (33,37), (37,49), (49,48), (48,54)
	(54,60)	(54,55), (55,56), (56,57), (57,60)
	(60,20)	(60,61), (61,55), (55,54), (54,53), (53,52), (52,50), (50,51), (51,35), (35,20)
	(20,49)	(20,19), (19,21), (21,34), (34,36), (36,37), (37,49)
	(49,29)	(49,48), (48,47), (47,46), (46,45), (45,41), (41,42), (42,29)
	(29,0)	(29,28), (28,12), (12,0)
2	(4,11)	(4,3), (3,2), (2,8), (8,9), (9,10), (10,11)
	(11,45)	(11,13), (13,27), (27,26), (26,30), (30,41), (41,45)
	(45,36)	(45,46), (46,47), (47,48), (48,49), (49,37), (37,36)
	(36,28)	(36,37), (37,49), (49,48), (48,47), (47,46), (46,45), (45,41), (41,42), (42,29), (29,28)
	(28,23)	(28,29), (29,27), (27,26), (26,14), (14,15), (15,16), (16,24), (24,23)
	(23,16)	(23,24), (24,16)
	(16,4)	(16,17), (17,7), (7,6), (6,3), (3,4)
3	(63,42)	(63,62), (62,53), (53,54), (54,55), (55,56), (56,57), (57,58), (58,44), (44,42)
	(42,10)	(42,29), (29,27), (27,13), (13,11), (11,10)
	(10,62)	(10,11), (11,13), (13,27), (27,29), (29,42), (42,44), (44,58), (58,57), (57,56), (56,55), (55,61), (61,62)
	(62,9)	(62,53), (53,54), (54,55), (55,56), (56,46), (46,45), (45,41), (41,30), (30,26), (26,14), (14,15), (15,9)
	(9,40)	(9,15), (15,16), (16,24), (24,25), (25,31), (31,40)
	(40,51)	(40,39), (39,47), (47,48), (48,49), (49,37), (37,36), (36,50), (50,51)
	(51,63)	(51,63)
4	(59,52)	(59,60), (60 61), (61,55), (55,54), (54,53), (53,52)
	(52,3)	(52,50), (50,36), (36,34), (34,21), (21,19), (19,5), (5,6), (6,3)
	(3,42)	(3,6), (6,7), (7,17), (17,16), (16,15), (15,14), (14,26), (26,27), (27,29), (29,42)
	(42,32)	(42,41), (41,30), (30,31), (31,25), (25,24), (24,23), (23,32)
	(32,19)	(32,33), (33,22), (22,21), (21,19)
	(19,10)	(19,5), (5,6), (6,7), (7,8), (8,9), (9,10)
	(10,59)	(10,11), (11,13), (13,12), (12,28), (28,43), (43,59)