Interview with Amadeus

Mohammad Mahdi Ahmadian

PhD student, University of Technology Sydney

MohammadMahdi.Ahmadian@student.uts.edu.au

November 1, 2021

Overview

Pre-processing rule no. 1

- If$(P_i - R_i \geq (D - D_i) \times G_i)$
  - Never purchase machine i
- In other words charges of machine is more than its profit;
- For example, for the third instance we have:
- $10 - 5 \geq (30 - 30) \times 3$

Pre-processing rule no. 2

- For any pair of machines i and j
- If$(D_i = D_j \& G_i = G_j \& R_i < R_j)$
  - Never purchase machine i
- This rule can be easily applied for fifth and sixth instances;
- 1 10 4 3
- 1 10 9 3

Calculating lower bound

Algorithm 1: Calculating the lower bound.

Input: *lower* $= C$.

for $i = 1$ **to** n **do**
- $temp = 0$
- **if** $P_i < C$ **then**
  - $temp = (D - D_i) \times G_i - P_i + R_i + C$;
  - **if** $lower < temp$ **then**
    - $lower := temp$;
  - **end**
- **end**

end

return *lower*;

Calculating upper bound

Algorithm 2: Calculating the upper bound.

Input: Let G_{max}, R_{max} and P_{min} be the maximum daily profit, maximum resale price and minimum machine price among machines available for sale in the future periods, $ub = 0$, M denote the set of machines owned by CVCM, *day* denote a given date and *prof* represent the monetary position of current node.

if *M is not empty* **then**

$ub = (D - day) \times G_{max} + R_{max} - P_{min} + prof$;

end

else

$ub = (D - M.front().D) \times \max(G_{max}, M.front().G) + prof$;

end

return *ub;*

Test case 1

	N	C	D	
	6	10	20	
i	D_i	P_i	R_i	G_i
1	6	12	1	3
2	1	9	1	2
3	3	2	1	2
4	8	20	5	4
5	4	11	7	4
6	2	10	9	1

M=Ø

Test case 1: Lower bound

$temp = (D - D_i) \times G_i - P_i + R_i + C;$
Lower bound $= (20 - 3) \times 2 - 2 + 1 + 10 = 43$

	N	C	D	
	6	10	20	
i	D_i	P_i	R_i	G_i
1	6	12	1	3
2	1	9	1	2
3	3	2	1	2
4	8	20	5	4
5	4	11	7	4
6	2	10	9	1

Test case 1: Upper bound

- $G_{max} = \max\{3, 2, 2, 4, 4, 1\} = 4$
- $R_{max} = \max\{1, 1, 1, 5, 7, 9\} = 9$
- $P_{min} = \min\{12, 9, 2, 20, 11, 10\} = 2$
- $prof = C = 10$
- $ub = (D - day) \times G_{max} + R_{max} - P_{min} + prof = (20 - 0) \times 4 + 9 - 2 + 10 = 97$

	N	C	D	
	6	10	20	
i	D_i	P_i	R_i	G_i
1	6	12	1	3
2	1	9	1	2
3	3	2	1	2
4	8	20	5	4
5	4	11	7	4
6	2	10	9	1

Test case 1

	N	C	D	
	6	10	20	
i	D_i	P_i	R_i	G_i
1	6	12	1	3
2	1	9	1	2
3	3	2	1	2
4	8	20	5	4
5	4	11	7	4
6	2	10	9	1

prof = 10

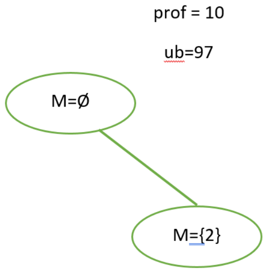
ub=97

M=∅

Test case 1: Monetary position

prof $= 10 - 9 + 1 = 2$

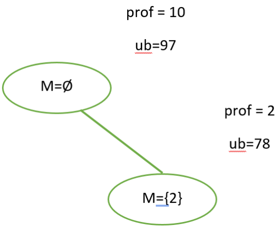
	N	C	D	
	6	10	20	
i	D_i	P_i	R_i	G_i
1	6	12	1	3
2	1	9	1	2
3	3	2	1	2
4	8	20	5	4
5	4	11	7	4
6	2	10	9	1



Test case 1: Upper bound

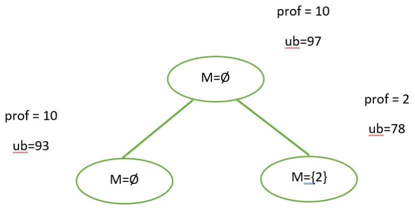
- $G_{max} = \max\{3, 2, 4, 4, 1\} = 4$
- prof = 10 - 9 + 1 = 2
- $ub = (D - M.front().D) \times \max(G_{max}, M.front().G) + prof = (20 - 1) \times \max(4, 2) + 2 = 78$;

	N	C	D	
	6	10	20	
i	D_i	P_i	R_i	G_i
1	6	12	1	3
2	1	9	1	2
3	3	2	1	2
4	8	20	5	4
5	4	11	7	4
6	2	10	9	1



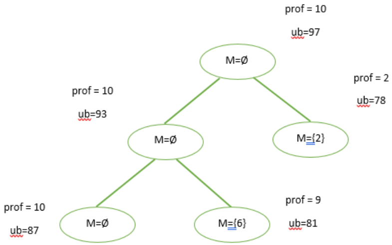
Test case 1

	N	C	D	
	6	10	20	
i	D_i	P_i	R_i	G_i
1	6	12	1	3
2	1	9	1	2
3	3	2	1	2
4	8	20	5	4
5	4	11	7	4
6	2	10	9	1



Test case 1

	N	C	D	
	6	10	20	
i	D_i	P_i	R_i	G_i
1	6	12	1	3
2	1	9	1	2
3	3	2	1	2
4	8	20	5	4
5	4	11	7	4
6	2	10	9	1



Output

Here is the output for 6 instances:

Case 1: 44

Case 2: 11

Case 3: 12

Case 4: 10

Case 5: 39

Case 6: 39

Just-In-Time

- Just-In-Time (JIT) is a production and inventory control system aiming at reducing inventory costs by purchasing materials or manufacturing products only when they are needed;
- Late deliveries may result in contractual penalties, loss of customer goodwill or losing future bidding opportunities.
- On the other hand early jobs can incur higher work-in-process or finished goods inventory levels which require more storage capacity.
- In scheduling theory such goals are often reflected by a number of performance criteria such as minimization of earliness and tardiness.
- My PhD research contributes to developing efficient algorithms for Just-In-Time (JIT) machine and shop scheduling problems

Traditional Manipulation Techniques

- Owning to the complexity of the problems, many authors have resorted to heuristics or meta-heuristics as an alternative to exact methods to address large instances.
- By manipulating the sequence new (improved) sequences can be generated.
- This optimization process has two major shortcomings:
- First, because the manipulations are mostly performed either randomly or myopically, i.e., without considering their impacts on the rest of the sequence, they usually fail to guide the algorithm towards generating high-quality sequences.
- Second, the function of the solver is relegated to scheduling only, i.e., the solver is only used to find the optimal operation completion times for a given sequence, so it makes no contribution to improving the sequence.

Overview of Relax and Solve

Algorithm 3: The relax-and-solve (R&S) matheuristic algorithm.

Input: An initial sequence Π.

```
while the stopping condition is not met do
    Relax();
    Solve();
end
return The best obtained sequence;
```

Formal Description of Relax and Solve

- The formation of a sub-sequence is controlled by the two parameters of "relaxation center" ($RC \in \mathbb{Z}^+$) and "relaxation radius" ($RR \in \mathbb{Z}^+$).
- While RC determines the centre position of the sub-sequence, i.e., the operation positioned in the middle of the sub-sequence, RR specifies the size of the sub-sequence. Therefore, a sub-sequence $\tilde{\Pi} \subset \Pi$ is formed by fixing its middle position, which is determined by RC, and certain positions before and after the middle position, which are determined by RR. The process of relaxation neighborhood is illustrated in Figure 1.

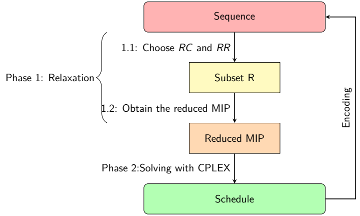


Figure 1: The global process of relaxation neighborhoods.

Aircraft Landing Problem

- Problems such as Aircraft Landing Problem (ALP) (Beasley et al., 2000) are concerned with air traffic control;
- The ALP aims to schedule aircraft landings such that the total deviation from target arrival times is minimized;
- Introduced by (Beasley et al., 2000), ALP deals with inbound traffic, in which each aircraft has a target landing time. There is also a separation time between landing/taking-off of each pair of aircraft which depends on their weight category (Light, Medium or Heavy). Loss of separation may result in collision;

Just-In-Time Job Shop Scheduling Problem

- Generally speaking, the job-shop scheduling models considered in the context of JIT manufacturing can be classified into two categories, namely those with due-dates on the job level and those with due-dates on the operation level;
- Just-in-time job shop scheduling (JIT-JSS) is a variant of the job shop scheduling problem, in which each operation has a distinct due date and any deviation of the operation completion time from its due date incurs an earliness or tardiness penalty (Baptiste et al., 2008);

Ongoing Project 1: Robust Aircraft Scheduling Problem

- The Aircraft Scheduling Problem (ASP) Furini et al. (2015), also known as the runway scheduling problem, aims to improve runway utilization by optimally assigning aircraft to the runway and scheduling the departure and arrival operations of the runway such that the total (weighted) delay in landing and take-off operations is minimized;
- We investigate proactive-reactive response strategies which can be obtained by extensions of robust optimization such as “Recoverable Robustness” (Liebchen et al., 2009);
- I am going to model this problem as Two-stage Robust Optimization. Assume that the ASP is defined in two stages such that we are required to find a first-stage solution that should be robust against the possible realizations (scenarios) of the input data in a second stage (i.e. disruptions caused by e.g. bad weather). This means that the first-stage solution is expected to perform reasonably well, in terms of feasibility and/or optimality, for any possible realization of the uncertain parameters.

Ongoing Project 2: An optimization framework for aircraft noise abatement

- The uneven spread of noise in proximity to an airport area can be regarded as a fairness dilemma: the noise has to be shouldered by one group, while the potential advantages of the airport are shared by others;
- A major method for noise abatement is assignment of preferred runways. However, most studies simply overlook runway allocation and only focus on design and selection of aircraft departure routes and the allocation of flights among these routes. As a result I am trying to propose a cross-layer proportional fairness framework to jointly optimize, rout assignment and runway allocation;

References

Baptiste, P., M. Flamini, and F. Sourd (2008). "Lagrangian bounds for just-in-time job-shop scheduling". In: *Computers & Operations Research* 35(3). Part Special Issue: New Trends in Locational Analysis, pp. 906 –915. ISSN: 0305-0548.

Beasley, J. E., M. Krishnamoorthy, Y. M. Sharaiha, and D Abramson (2000). "Scheduling aircraft landings—the static case". In: *Transportation science* 34(2), pp. 180–197.

Furini, F., M. P. Kidd, C. A. Persiani, and P. Toth (2015). "Improved rolling horizon approaches to the aircraft sequencing problem". In: *Journal of Scheduling* 18(5), pp. 435–447. ISSN: 1099-1425.

Liebchen, C., M. Lübbecke, R. Möhring, and S. Stiller (2009). "The concept of recoverable robustness, linear programming recovery, and railway applications". In: *Robust and online large-scale optimization*. Springer, pp. 1–27.