Advanced Models and Methods in Operations Research
Project: Batch scheduling

Florian Fontan

2021–2022

For each problem considered, instances and a code skeleton containing an instance parser and a solution checker are provided in the `data/` and `python/` folders of the project.

The algorithms must be implemented in the provided files between the tags `TODO START` and `TODO END`.

They must be tested on all the provided instances with the command: `python3 problem.py -i instance.json -c certificate.json`

And each solution file must be validated by the provided checker: `python3 problem.py -a checker -i instance.json -c certificate.json`

The results must be reproducible.

The delivrable must contain:

- A *short* report describing and justifying the proposed algorithms
- The code implementing the algorithms
- The solution files obtained on the provided instances

1 Dynamic Programming

We consider the Knapsack problem with width:

- Input:
  - n items; for each item $j = 1, \dots, n$
    - a weight $w_j \in \mathbf{N}^+$
    - a width $l_j \in \mathbf{N}^+$
    - a profit $p_j \in \mathbf{N}^+$
  - a capacity $C \in \mathbf{N}^+$
- Problem: select a subset of items such that
  - the total weight of the selected items does not exceed the knapsack capacity
- Objective: maximize the total profit of the selected items minus the maximum width among the selected items

Propose and implement an algorithm based on Dynamic Programming for this problem.

2 Heuristic Tree Search

We consider the Knapsack problem with width and conflicts:

- Input:
  - n items; for each item $j = 1, \dots, n$
    - a weight $w_j \in \mathbf{N}^+$
    - a width $l_j \in \mathbf{N}^+$
    - a profit $p_j \in \mathbf{N}^+$
  - a capacity $C \in \mathbf{N}^+$
  - a graph G such that each node corresponds to an item
- Problem: select a subset of items such that
  - the total weight of the selected items does not exceed the knapsack capacity
  - if there exists an edge between vertex j_1 and j_2 in G, then item j_1 and item j_2 must not be both selected
- Objective: maximize the total profit of the selected items minus the maximum width among the selected items

Propose and implement an algorithm based on Heuristic Tree Search with Dynamic Programming for this problem.

3 Column Generation + Dynamic Programming

We consider the Single machine batch scheduling problem with Makespan objective:

- Input:
  - n jobs; for each job $j = 1, \ldots, n$, a processing time $p_j \in \mathbf{N}^+$ and a size $s_j \in \mathbf{N}^+$
  - a batch capacity $Q \in \mathbf{N}^+$
- Problem: partition the jobs into batches and sequence the batches such that:
  - each job must be in exactly one of the batches
  - the processing time of a batch is equal to the longest processing time among all jobs it contains
  - the total size of the jobs in a batch does not exceed its capacity
- Objective: minimize the makespan of the schedule

Propose an exponential formulation and implement an algorithm based on a Column Generation heuristic for this problem.

4 Column Generation + Heuristic Tree Search

We consider the Single machine batch scheduling problem with conflicts and Makespan objective:

- Input:
  - n jobs; for each job $j = 1, \ldots, n$, a processing time $p_j \in \mathbf{N}^+$ and a size $s_j \in \mathbf{N}^+$
  - a batch capacity $Q \in \mathbf{N}^+$
  - a graph G such that each node corresponds to a job
- Problem: partition the jobs into batches and sequence the batches such that:
  - each job must be in exactly one of the batches
  - the processing time of a batch is equal to the longest processing time among all jobs it contains
  - the total size of the jobs in a batch does not exceed its capacity
  - if there exists an edge between vertex j_1 and vertex j_2 in G, then job j_1 and job j_2 must not be in the same batch
- Objective: minimize the makespan of the schedule

Propose an exponential formulation and implement an algorithm based on a Column Generation heuristic for this problem.