Heuristic approach with machine learning alongside for the resource-constrained shortest path problem

Fulin Yan, François Clautiaux, Aurélien Froger, Boris Albar

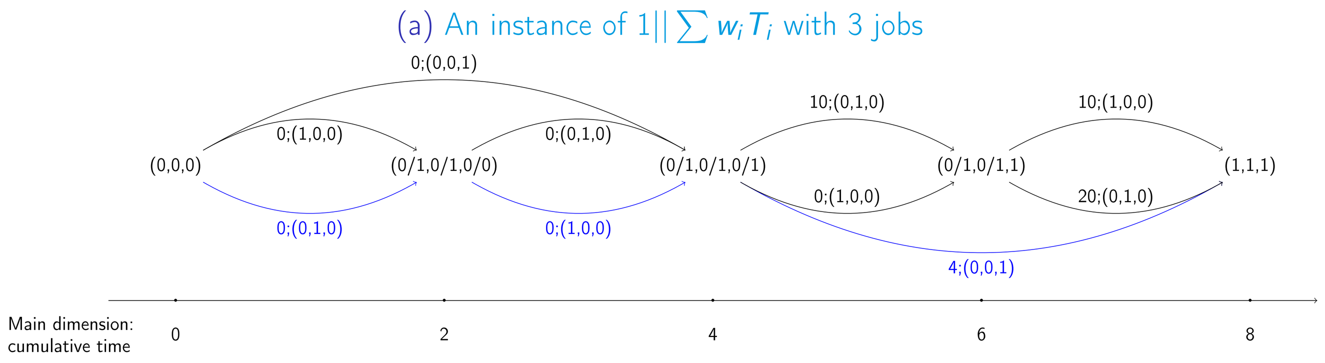
Centre Aquitain des Technologies de l'Information et Electroniques, F-33400 Talence, France

Problem introduction

- We study the problem that can be formulated partially or completely as a sequential decision process with additional linear constraints related to the resource consumption in the whole decision sequence.
  - Many combinatorial optimization problems can be expressed as sequential decision processes (SDP) such as planning, knapsack, scheduling, véhicule routing.
  - In the SDP formalism, a representation of the problem (or a sub-problem thereof) is associated with a acyclic directed multi-graph representing the states of the system and the possible transitions between these states.
  - Network-flow formulations [de Lima et al., 2022].
  - We study a special case of network formulations: resource-constrained shortest path problem (RCSPP).

Job	processing time	due time	weight
1	2	7	10
2	2	4	6
3	4	6	10

(a) An instance of $1||\sum w_i T_i$ with 3 jobs



(b) The associated graph

Figure: An example of reformulating one single machine weighted total tardiness problem $1||\sum w_i T_i$ as RCSPP

Research objectives

- The present study investigates the following objectives:
  - Develop a generic heuristic tree-search approach with machine learning models alongside to produce useful information for RCSPP.
  - Train machine learning models to produce useful information.
  - Compare the performance to classical tree search without learning and to problem-specific heuristics.

Arc-flow formulation for RCSPP

- $\mathcal{G}(\mathcal{V},\mathcal{A})$ be a directed acyclic multi-graph, where $s \in \mathcal{V}$ represents the source and $t \in \mathcal{V}$ represents the sink.
- Each arc $a \in \mathcal{A}$ has a tail $v^-(a)$ and a head $v^+(a)$.
- For each arc $a \in \mathcal{A}$, c_a represents the cost of a.
- Let $\mathcal{R}$ be the set of resource constraints. For each resource constraint indexed by $k \in \{1 \dots, |\mathcal{R}|\}$, we have the resource capacity denoted by r^k, and each arc a is also assigned a resource consumption r_a^k.
- A arc-flow formulation of RCSPP is given as follows in (1) - (4):

$$\min \sum_{a\in\mathcal{A}} c_a x_a \tag{1}$$

$$\text{s.t.} \sum_{a\in\mathcal{A}|v^+(a)=v} x_a - \sum_{a'\in\mathcal{A}|v^-(a')=v} x_{a'} = \begin{cases} -1, & \text{if } v = s,\\ 1, & \text{if } v = t,\\ 0, & \text{otherwise,}\end{cases} \quad \forall v \in \mathcal{V}, \tag{2}$$

$$\sum_{a\in\mathcal{A}} r_a^k x_a \le (\text{or} =) r^k, \quad \forall k \in \{1,\dots,|\mathcal{R}|\}, \tag{3}$$

$$x_a \in \{0,1\}, \quad \forall a \in \mathcal{A} \tag{4}$$

Beam search

- A constructive heuristic, which starts with an empty solution and repeatedly extends the current partial solution until a complete solution is obtained.
- Vocabulary:
  - $\ell \in \mathcal{L}$ label: a partial solution
  - β beam size: the maximum number of labels to keep at each level
  - $\psi : \mathcal{L} \to \mathbb{R}$ heuristic criterion: a function to score $\ell \in \mathcal{L}$.
  - g_ℓ: value of the partial solution ℓ
  - $\mathcal{B}^n \subset \mathcal{L}$: a set of labels at level n of the beam search.
  - $\mathcal{F}^n \subset \mathcal{L}$: a set of complete solutions at level n of the beam search
- Choice of ψ:
  - Many classical tree-search based algorithms use ν_ℓ the dual bound of solutions that complete ℓ as the score $\psi(\ell)$ [Kuroiwa and Beck, 2023].
  - The dual bound quality impact directly to the solution quality of the beam search.

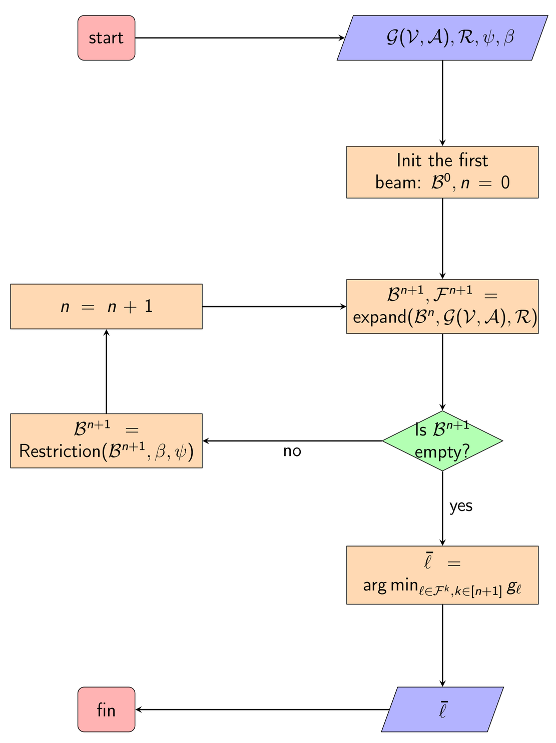


Figure: Flowchart of Beam search

Beam search with machine learning alongside

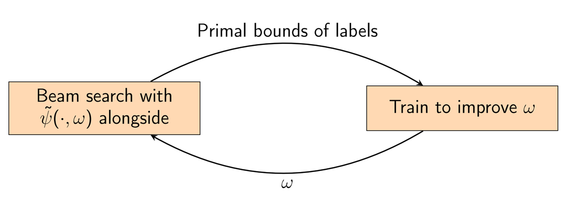
- Using a machine learning model to estimate the scoring function ψ.
- Let ϕ be a mapping from a label to a vector of m features representing the label.

$$\phi : \mathcal{L} \to \mathbb{R}^m$$
$$\ell \to \phi(\ell)$$

- Let $\tilde{\psi}(\cdot;\omega)$ be a function parameterized by ω such that $\psi(\ell;\omega) = \tilde{\psi}(\phi(\ell);\omega)$

$$\tilde{\psi}(\cdot;\omega) : \mathcal{R}^m \to \mathbb{R}$$
$$\phi(\ell) \to \psi_\ell(\omega) = \tilde{\psi}(\phi(\ell);\omega)$$

- The objective is to train $\tilde{\psi}(\cdot;\omega)$ such that $\psi_\ell(\omega)$ estimates the primal bound of solutions completing ℓ. [Huber and Raidl, 2022]



Exploration phase

- The exploration phase uses beam search to generate diverse labels with primal bounds.
- The generated labels should be diverse enough to improve the training efficiency.
- Generate as many labels as we can v.s. generate "good" labels?

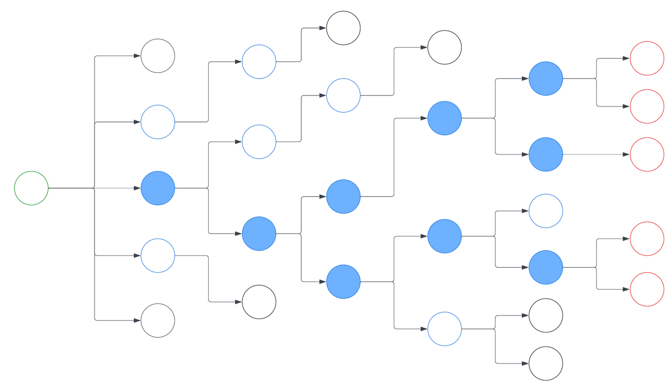


Figure: Lack of diversity for labels with primal bounds in beam search. An example with $\beta = 3$. Nodes represent labels. Nodes filled in blue are labels with primal bound generated by the beam search.

- Perform a greedy rollout on each label $\ell \in \mathcal{B}^{n+1}$
- If the completion of ℓ is optimal proven by its dual bound, the best primal bound of solutions completing ℓ is found. No need to further expand ℓ. Move label ℓ into $\mathcal{U}^{n+1}$.
- Labels in $\mathcal{U}^{n+1}$ are concatenated with the associated rollout solution and are added into $\mathcal{F}^{n+1}$.

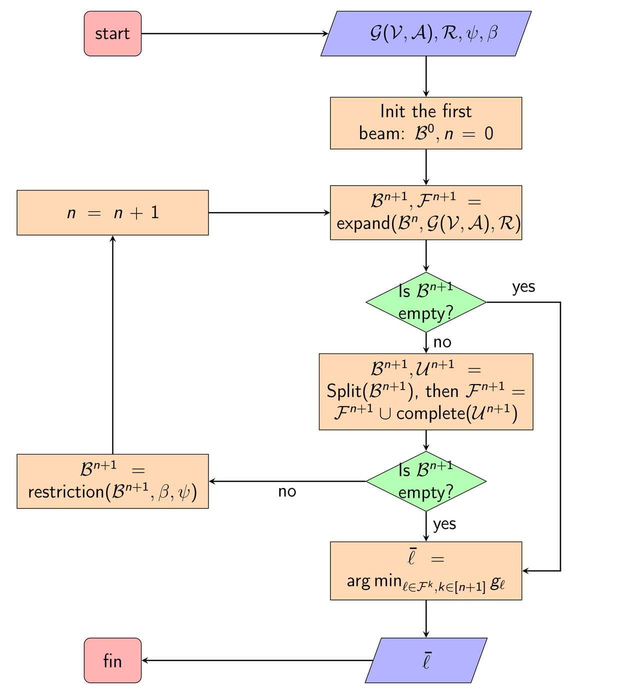


Figure: Flowchart of Beam search with greedy rollout

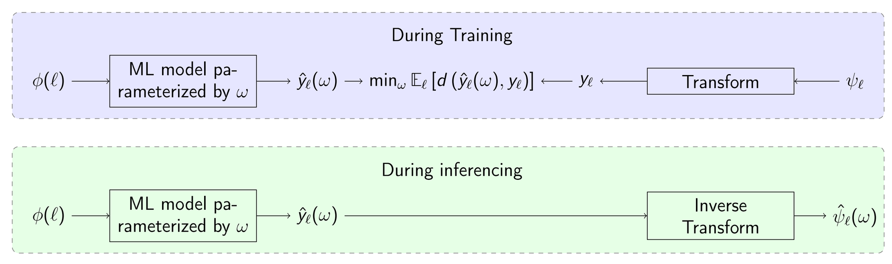
Features of labels

- The number of resources varies as the instance size varies.
- Cost of labels ($\boldsymbol{\lambda}^a = 0$)
- Cost of labels ($\boldsymbol{\lambda} = \boldsymbol{\lambda}^{*\,b}$)
- Dual completion of labels ($\boldsymbol{\lambda} = 0$)
- Dual completion of labels ($\boldsymbol{\lambda} = \boldsymbol{\lambda}^*$)
- Sum of resource consumption over $\boldsymbol{r}^T\boldsymbol{\lambda}^* > 0$
- Sum of resource consumption over $\boldsymbol{r}^T\boldsymbol{\lambda}^* < 0$
- Sum of resource consumption over $\boldsymbol{r}^T\boldsymbol{\lambda}^* \in [\min, \max]$
- Sum of resource consumption over a quantile for some indicator
- Sum of resource slack over ...
- Main dimension value
- ...

[a] $\boldsymbol{\lambda}$: the Lagrangian multipliers of the resource constraints (3)
[b] $\boldsymbol{\lambda}^*$: the best Lagrangian multipliers of constraints (3)

Transformations and inverse transformations

- The scale of ψ_ℓ primal bounds of labels varies between instances. Transforming them into the same scale is vital for the convergence of machine learning models.
- The transformation must be bijective.

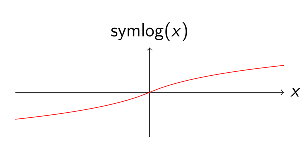


- Alternatives of transformations
  - Normalize by ν_ℓ the dual bound of the label

$$y_\ell = \frac{\nu_\ell}{\psi_\ell}, \quad \hat{\psi}_\ell(\omega) = \frac{\nu_\ell}{\hat{y}_\ell(\omega)}$$

  - Symlog transformation [Hafner et al., 2023]

$$\text{symlog}(x) = \text{sign}(x)\log(1+|x|), \quad \text{symexp}(y) = \text{sign}(y)(\exp(|y|) - 1)$$
$$y_\ell = \text{symlog}(\psi_\ell), \quad \psi_\ell(\omega) = \text{symexp}(\hat{y}_\ell(\omega))$$



Preliminary numerical results

- 125 instances of $1||\sum w_i T_i$ with 100 jobs.
- The preliminary results show the solution quality of beam search (BS) using ν_ℓ as heuristic criterion without ML model alongside compared to the baseline.
- Baseline: early due date (EDD) + shortest processing time (SPT) + local search (LS)

Heuristic	beam size	meanGap(%)	stdGap(%)
EDD+SPT+LS	-	-	-
EDD	-	55.11	18.52
SPT	-	62.42	29.40
BS	10	2.23	3.41
	20	1.57	2.67
	100	1.00	1.90
	200	0.86	1.76
	500	**0.66**	**1.39**

Conclusions

- Machine learning alongside heuristic methods remains a relatively novel research area, with limited achievements thus far.

Perspectives

- Converge machine learning models.
- Generalize models for $1||\sum w_i T_i$ problem.
- Study more complex problems like the disjunctive knapsack problem.

References

de Lima, V. L., Alves, C., Clautiaux, F., Iori, M., and de Carvalho, J. M. V. (2022).
Arc Flow Formulations Based on Dynamic Programming: Theoretical Foundations and Applications.
European Journal of Operational Research, 296(1):3–21.

Hafner, D., Pasukonis, J., Ba, J., and Lillicrap, T. (2023).
Mastering Diverse Domains through World Models.

Huber, M. and Raidl, G. R. (2022).
Learning Beam Search: Utilizing Machine Learning to Guide Beam Search for Solving Combinatorial Optimization Problems.
In Nicosia, G., Ojha, V., La Malfa, E., La Malfa, G., Jansen, G., Pardalos, P. M., Giuffrida, G., and Umeton, R., editors, *Machine Learning, Optimization, and Data Science*, Lecture Notes in Computer Science, pages 283–298. Springer International Publishing.

Kuroiwa, R. and Beck, J. C. (2023).
Solving Domain-Independent Dynamic Programming Problems with Anytime Heuristic Search.
Proceedings of the International Conference on Automated Planning and Scheduling, 33:245–253.

Academic partners