A combinatorial flow-based formulation for temporal bin packing problems

John Martinovic[⋆], Nico Strasdat[⋆], José Valério de Carvalho[†], Fabio Furini[‡]

[⋆] Institute of Numerical Mathematics, Technische Universität Dresden, Dresden, Germany
[†] Departamento de Produção e Sistemas/Centro ALGORITMI, Universidade do Minho, Braga, Portugal
[‡] Department of Computer, Control and Management Engineering "Antonio Ruberti", Sapienza University of Rome, Rome, Italy

vc@dps.uminho.pt

Column Generation 2023 - May 16-19

Contents

The temporal bin packing problem (TBPP) (informal)

- Assign *items* (*jobs*) with well-defined size, start and end times to *bins* (*servers*) with a given capacity, respecting capacity at any instant of time, with the objective of minimizing the number of bins.
- Example E_0 [Furini'06]:

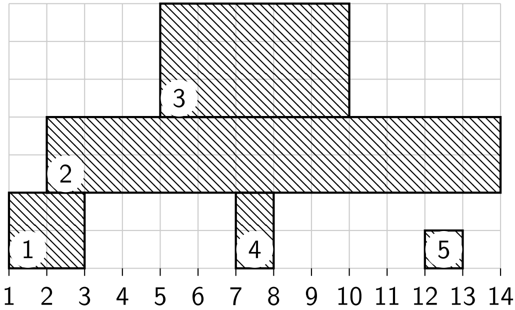


Figure: Horizontal axis: time instants; vertical grid: item sizes.

- If capacity of bins $C = 5$, optimal solution requires 2 bins.

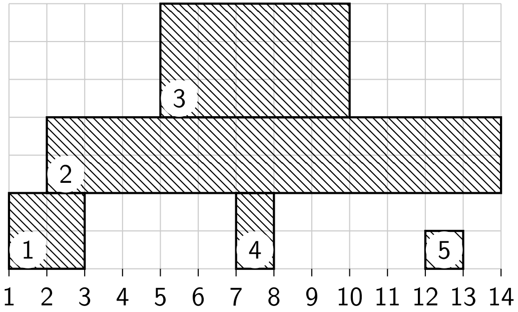
One application: data-center workload management

- 'One emerging management technique is to make sure that servers are at full throttle as much of the time as possible, whereas others are turned off rather than being left idle.' [Jones'2018]
- Cloud providers have to assign a set of tasks to physical machines.
- Problem coined as the Temporal Bin-Packing Problem (TBPP) by:

  de Cauwer, M., Mehta, D., O'Sullivan, B., 2016. The temporal bin packing problem: an application to workload management in data centres. In: Proceedings of the 28$^{\text{th}}$ IEEE International Conference on Tools with Artificial Intelligence, pp. 157-164.
- It is a hard combinatorial problem.

A closely related problem: the TBPP with Fire-Ups

- TBPP-FU: minimize weighted sum of number of servers in operation and number of fire-ups.



Two solutions with 2 bins ($C = 5$), but a different number of fire-ups:

1. Assigning $\begin{cases} \{2,3\} & \rightarrow & \text{Bin}_1 \\ \{1,4,5\} & \rightarrow & \text{Bin}_2 \end{cases}$ leads to $1 + 3 = 4$ fire-ups.
2. Assigning $\begin{cases} \{3\} & \rightarrow & \text{Bin}_1 \\ \{1,2,4,5\} & \rightarrow & \text{Bin}_2 \end{cases}$ leads to $1 + 1 = 2$ fire-ups.

Other related problems

- *temporal knapsack problem (TKP)*, special case with a single server (*e.g.*, Bartlett *et al.* '05, Caprara *et al.* '13, de Cauwer *et al.* '16, Gschwind *et al.* '17, Aydin *et al.* '20, Clautiaux *et al.* '21)
- *interval scheduling with a resource constraint* (*e.g.*, Angelelli *et al.* '14)
- *bandwidth allocation problem (BAP)* (*e.g.*, Bar-Noy *et al.* '99, Chen *et al.* '02)
- *storage allocation problem (SAP)* (*e.g.*, Chen *et al.* '02)
- *dynamic bin packing problem* (*e.g.*, Coffman *et al.* '83)

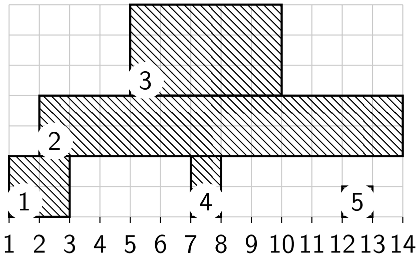
Other research in TBPP and TBPP-FU

- Dell'Amico, M., Furini, F., Iori, M., 2020. A branch-and-price algorithm for the temporal bin packing problem, Computers & Operations Research 114, 104825
- Aydin, N., Muter, I., Birbil, S.I., 2020. Multi-objective temporal bin packing problem: an application in cloud computing. Computers & Operations Research 121, 104959.
- Martinovic, J., Strasdat, N., Selch, M., 2021. Compact integer linear programming formulations for the temporal bin packing problem with fire-ups. Computers & Operations Research 132, Article 105288.
- Martinovic, J., Strasdat, N., VC, Furini, F., 2022. Variable and constraint reduction techniques for the temporal bin packing problem with fire-ups. Optimization Letters 16, 2333-2358.
- Martinovic, J., Strasdat, N., 2022. Theoretical insights and a new class of valid inequalities for the temporal bin packing problem with fire-ups. Preprint MATH-NM-01-2022, Technische Universität Dresden.

TBPP: terminology

- Assign a set of n jobs, each with a resource consumption c_i and an activity interval $[s_i, e_i)$ (*starting time* and *ending time*) with $s_i < e_i$, $i \in I := \{1, \dots, n\}$,
- to a sufficiently large number of homogeneous *bins* of *capacity* C.
- Instance is defined by $E = (n, C, \boldsymbol{c}, \boldsymbol{s}, \boldsymbol{e})$, where $\boldsymbol{c}, \boldsymbol{s}, \boldsymbol{e}$ are vectors of item sizes, starting times and ending times, resp..
- All input data are nonnegative integer numbers.

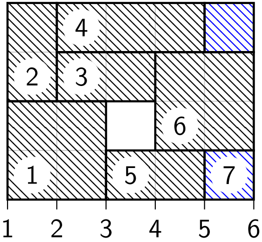
Example [Furini'06]: $E_0 = (5, 5, (2, 2, 3, 2, 1), (1, 2, 5, 7, 12), (3, 14, 10, 8, 13))$



- $T = \{1, 2, 3, 5, 7, 8, 10, 12, 13, 14\}$: set of times
- $T_S = \{1, 2, 5, 7, 12\}$: set of starting times

TBPP and TBPP-FU and related problems: a closer look

- A packing is feasible for the TBPP and TBPP-FU if capacity is enough; items do not have to be represented as connected objects:



- This packing is feasible for bandwidth allocation in wireless networks [Chen *et al.* '02].
- It is not feasible for the strip packing problem (SPP): there is no rearrangement of the items that enables representing the blue item $i = 7$ with $[s_i, e_i) = [5, 6)$ and $c_i = 2$ as a single rectangle of size 1×2 without destroying the rectangular structure of another item.

Compact Model for the TBPP [Dell'Amico et al.'20]

$$z^{com} = \sum_{k \in K} z_k \rightarrow \min$$

$$\text{s.t.} \quad \sum_{k \in K} x_{ik} = 1, \qquad i \in I, \tag{1}$$

$$\sum_{i \in I_t} c_i x_{ik} \leq C \cdot z_k, \qquad t \in T, k \in K, \tag{2}$$

$$x_{ik} \in \{0,1\}, \qquad i \in I, k \in K, \tag{3}$$

$$z_k \in \{0,1\}, \qquad k \in K. \tag{4}$$

- $I := \{1, \ldots, n\}$: the set of items;
- $K := \{1, \ldots, n\}$: the set of all servers (n is an upper bound);
- $z_k = 1$ if server $k \in K$ is used; 0, otherwise;
- $x_{ik} = 1$ if item $i \in I$ is assigned to server $k \in K$; 0, otherwise.

Exponential-size model for the TBPP [Dell'Amico *et al.* '20]

$$z^{exp} = \sum_{j \in \mathscr{J}} \xi_j \rightarrow \min$$

$$\text{s.t.} \quad \sum_{j \in \mathscr{J}} x_i^j \xi_j = 1, \qquad i \in I, \qquad (5)$$

$$\xi_j \in \{0,1\}, \qquad j \in \mathscr{J}. \qquad (6)$$

where:

- $\mathscr{J}$: index set of $\mathscr{P}$ (the set of patterns).
- ξ_j : states whether pattern j is used $(\xi_j = 1)$ or not $(\xi_j = 0), j \in \mathscr{J}$.
- x_i^j : element of n-dimensional incidence vector $\boldsymbol{x}^j \in \{0,1\}^n$ that tells whether item $i \in I$ is contained in pattern j or not, $j \in \mathscr{J}$.

- A branch-and-price algorithm (B&P$^+$) that uses `Cplex` to solve a temporal knapsack (TKP) subproblem is presented in:

[23] Dell'Amico, M., Furini, F., Iori, M.: A branch-and-price algorithm for the temporal bin packing problem. Computers & Operations Research 114, Article 104825 (2020)

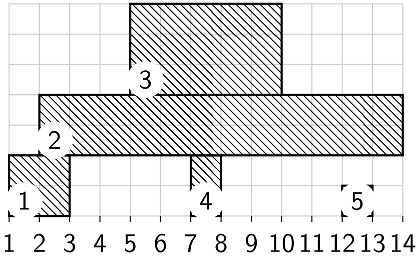
The temporal knapsack (TKP) subproblem

- Subproblem can be solved as a multidimensional knapsack problem.

Any feasible assignment of jobs to a single server is called *a pattern*:

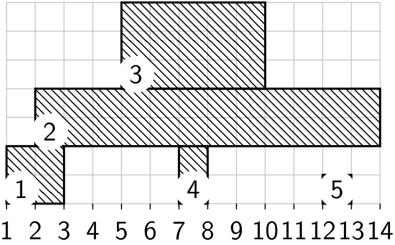
$$\text{Set of patterns } \mathscr{P} := \left\{ \boldsymbol{x} \in \{0,1\}^n \,\middle|\, \sum_{i \in I_t} c_i x_i \le C,\, t \in T \right\},$$

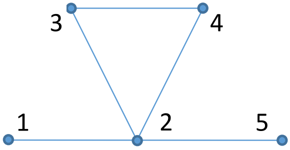
- where I_t : set of items active at time t (*e.g.*, $I_7 = \{2,3,4\}$).
- note: T can also be replaced by T_S^{nd}, the set of (non-dominated) starting times.



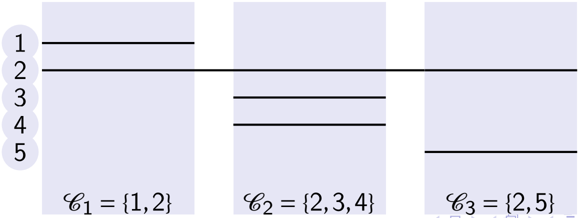
Property: interval graph representation

- Graph with edges of time-intersecting items is an *interval graph*.





- The maximal cliques of an interval graph G can be linearly ordered such that, for every vertex x of G, the maximal cliques containing x occur consecutively [Gilmore and Hoffman'64].

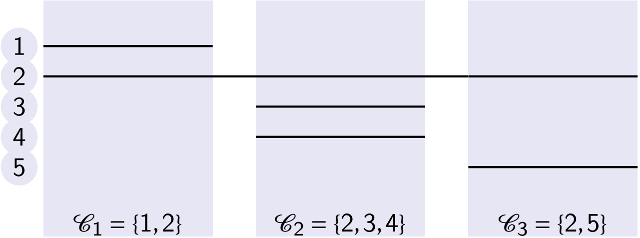


TKP subproblem can be solved with dynamic programming

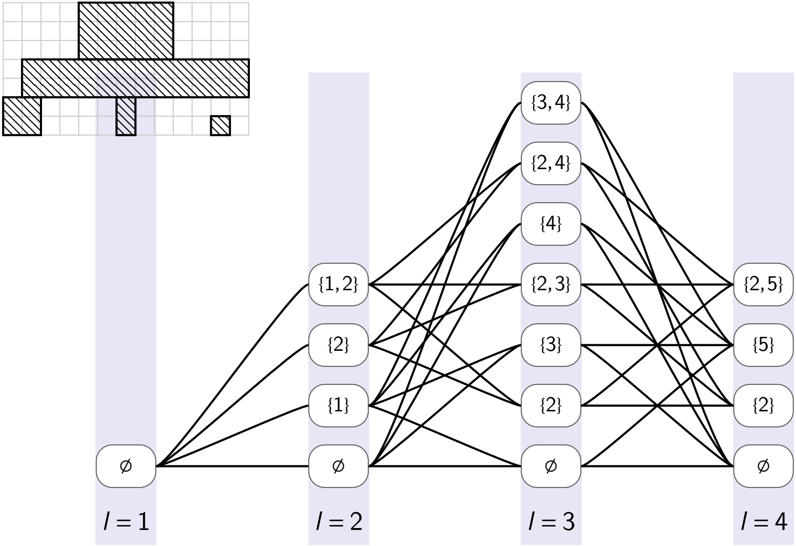
- problem structured in stages (layers), each corresponding to a clique.

(exponential size) state space of a given clique $\mathscr{C}_j$ (layer$_j$)

- state: subset of items of clique $\mathscr{C}_j$ that fit in a bin.
- its label includes space occupied by those items: this info is needed to determine allowed decisions (state transitions).



Combinatorial arc flow graph for TKP [Furini'06]



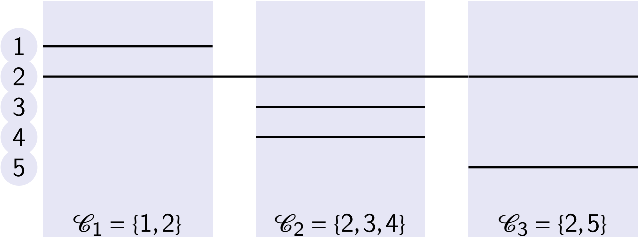
- A path in graph is a temporal knapsack (TKP) solution (a pattern).

Contribution

- For a given stage, after closing the items that do not belong to next clique, the set of items of different states becomes exactly the same.
- Shrink those 'equivalent' states into a *representative state.*

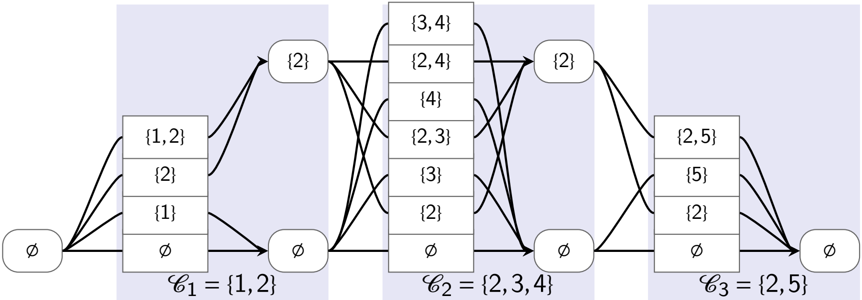
NEW (exponential size) state space of a given clique $\mathscr{C}_j$ (layer$_j$)

- state: subset of the items in the white space that fit in a bin.
- its label includes space occupied by those items: there is no loss of info required to take the decisions.



- Size of state space is significantly reduced.

(multi-)Graph for example E_0



- Multiple arcs between each pair of nodes are allowed.

Combinatorial Arc flow (CAF) model for the TBPP

- It is coined *Combinatorial*, because number of states is exponential.

$$z^{comb} = \sum_{e \in \mathcal{E}^{out}(0,\emptyset)} \xi_e \to \min$$

$$\text{s.t.} \quad \sum_{e \in \mathcal{E}^{in}(l,\widetilde{J})} \xi_e = \sum_{e \in \mathcal{E}^{out}(l,\widetilde{J})} \xi_e, \qquad (l,\widetilde{J}) \in \mathcal{V} \setminus \{(0,\emptyset),(m,\emptyset)\}, \tag{7}$$

$$\sum_{e \in \mathcal{E}(i)} \xi_e = 1, \qquad i \in I, \tag{8}$$

$$\xi_e \in \mathbb{Z}_+, \qquad e \in \mathcal{E}_l, l \in L. \tag{9}$$

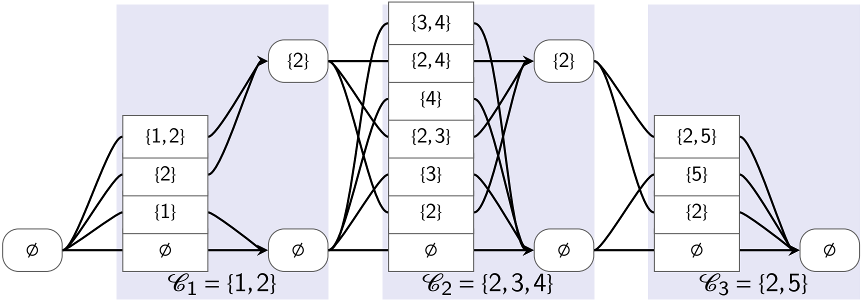
where

- $\xi_{l-1,l,J} \in \mathbb{Z}_+$: units of flow carried by an arc $(l-1,l,J) \in \mathcal{E}_l$, $l \in L$.
- $\mathcal{E}^{in}(l,J)$ and $\mathcal{E}^{out}(l,J)$: arcs entering and leaving a given state $(l,J) \in \mathcal{V}$, resp..

- Multiple arcs between each pair of nodes in network constraints (7).
- Arc can contribute to more than one side constraint, in (8).

Arcflow models for hard problems

- Find minimum flow in acyclic graph, ensuring all items are placed.
- Decompose optimal flow in paths to find patterns.



- Arc flow models have strong relaxations, comparable to those of strong models solved with column generation/lagrangean relaxation.

[20] de Lima, V.L., Alves, C., Clautiaux, F., Iori, M., VC: Arc flow formulations based on dynamic programming: theoretical foundations and applications. European Journal of Operational Research 296(1), 3-21 (2022)

TBPP-FU: compact M1-type model [Aydin *et al.* '2020]

$$z^{(1)} = \gamma \cdot \sum_{k \in K} \sum_{t \in T_S} w_{tk} + \sum_{k \in K} z_k \to \min$$

$$\text{s.t.} \quad y_{tk} \leq \sum_{i \in I_t} c_i \cdot x_{ik} \leq y_{tk} \cdot C, \qquad k \in K, t \in T, \tag{10}$$

$$\sum_{k \in K} x_{ik} = 1, \qquad i \in I, \tag{11}$$

$$x_{ik} \leq y_{s_i,k}, \qquad i \in I, k \in K, \tag{12}$$

$$y_{tk} \leq z_k, \qquad k \in K, t \in T, \tag{13}$$

$$y_{tk} - y_{t-1,k} \leq w_{tk}, \qquad k \in K, t \in T_S, \tag{14}$$

$$x_{ik} \in \{0,1\}, \qquad i \in I, k \in K, \tag{15}$$

$$y_{tk} \in \{0,1\}, \qquad k \in K, t \in T, \tag{16}$$

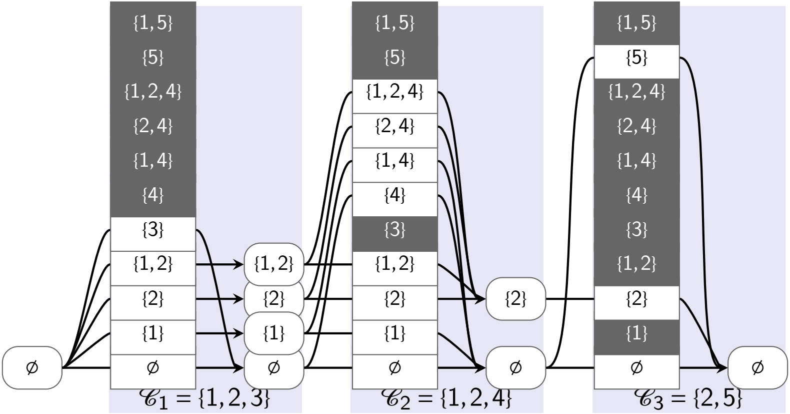
$$w_{tk} \geq 0, \qquad k \in K, t \in T_S, \tag{17}$$

$$z_k \in \{0,1\}, \qquad k \in K. \tag{18}$$

where

- $y_{tk} = 1$: represents a positive load on server k at time t.
- $w_{tk} = 1$: server $k \in K$ was activated at time $t \in T_S$.

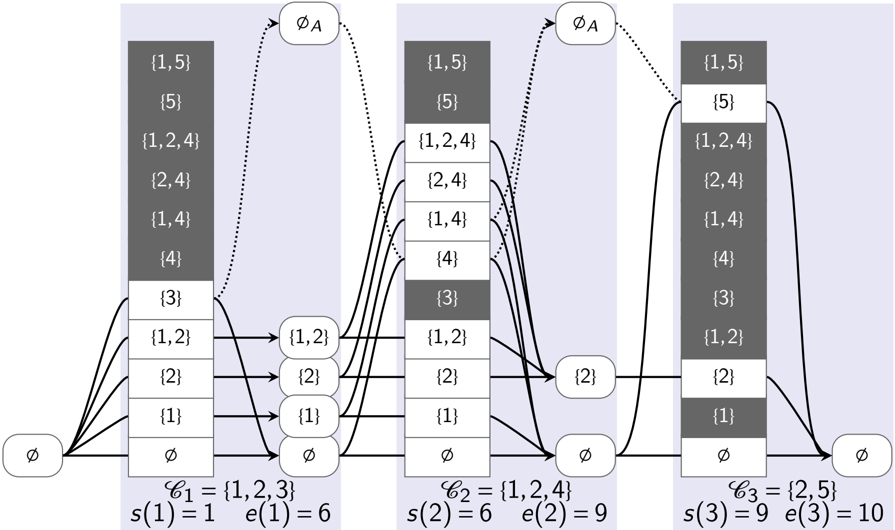
Instance $E_2 := (5,5,(1,2,5,2,4),(1,2,5,6,9),(7,10,6,9,10))$



note about this graph for the TBPP:

- the entire pattern set $\mathscr{P}$ in any layer is depicted.
- elements that are infeasible because they do not use the items allowed for the respective clique are colored black.

Instance E_2 : a glimpse at changes to model the TBPP-FU



Instances

Category (A) [Aydin *et al.* '20]

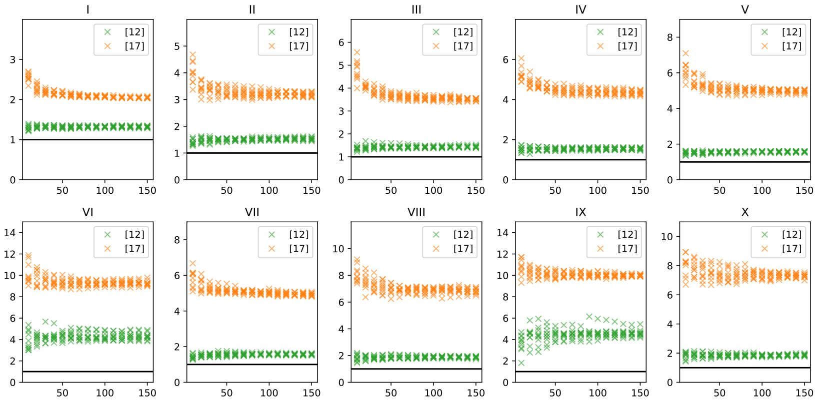
1. set of 240 instances: 48 different groups of 5 instances each;
2. all instances with $C = 100$, $\gamma = 1$. Groups differ in:
   - number of items: $n \in \{50, 100, 150, 200, 500, 1000\}$,
   - time horizon: dense scenario ($\bar{s} := \max_{i \in I}\{s_i\} = n$) vs. relaxed scenario ($\bar{s} = 1.2n$),
   - job duration: short 'd_S', $(e_i - s_i) \sim u[10, 30]$ vs. long 'd_L', $(e_i - s_i) \sim u[20, 60]$,
   - capacity consumption: low 'c_L', $c_i \sim u[25, 50]$ vs. high 'c_H', $c_i \sim u[25, 75]$.

(A1) set of 160 instances with values $n \leq 200$.
(A2) set of 80 significantly more difficult instances with $n \in \{500, 1000\}$.

Category (B) [Dell'Amico *et al.* '20]

1. set of 1500 instances: 15 groups of 10 classes (each with 10 inst.)
2. all instances with $C = 100$. Groups differ in:
   - number of non-dominated starting times (maximal cliques), *i.e.*, $|T_S^{nd}| \in \{10, 20, 30, \ldots, 150\}$

Graph size: number of CAF graph arcs scaled to 1



- Size of graph for instances of Category B is much smaller than in:

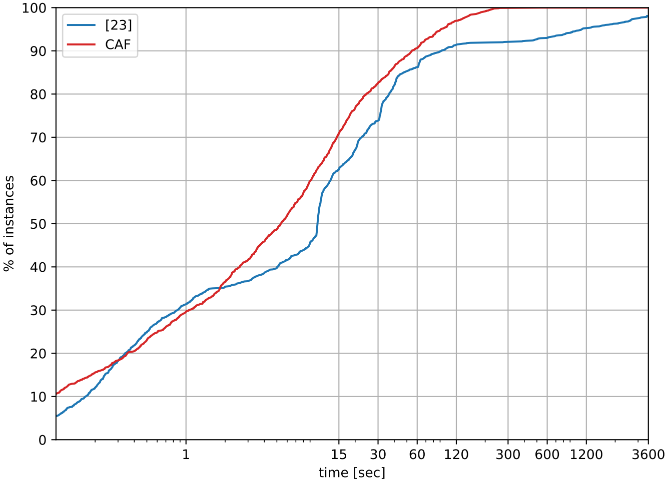
[12] Caprara, A., Furini, F., Malaguti, E.: Uncommon Dantzig-Wolfe reformulation for the temporal knapsack problem. INFORMS Journal on Computing 25(3), 560-571 (2013)
[17] Clautiaux, F., Detienne, B., Guillot, G.: An iterative dynamic programming approach for the temporal knapsack problem. European Journal of Operational Research 293(2), 442-456 (2021)

TBPP: comparison with B&P$^+$ for Category (B)

- B&P$^+$ (based on branch-and-price) is the best solution approach from the literature [23]:

$\lvert T_S^{nd} \rvert$	CAF with $t_{\max} = 1800s$ t	opt	[23] with $t_{\max} = 3600s$ t	opt
10	**0.4**	**(100)**	2.0	(100)
20	**1.4**	**(100)**	5.2	(100)
30	**2.5**	**(100)**	9.3	(100)
40	**3.7**	**(100)**	15.7	(100)
50	**6.9**	**(100)**	27.9	(100)
60	**10.0**	**(100)**	63.9	(100)
70	**15.7**	**(100)**	115.1	(100)
80	**16.5**	**(100)**	132.6	(100)
90	**20.4**	**(100)**	151.8	(99)
100	**23.9**	**(100)**	168.8	(99)
110	**26.7**	**(100)**	218.0	(98)
120	**30.7**	**(100)**	237.9	(99)
130	**38.6**	**(100)**	432.9	(94)
140	**40.2**	**(100)**	475.1	(92)
150	**51.5**	**(100)**	559.4	(90)
Average	**19.3**	**(1500)**	174.4	(1471)

TBPP: performance profile: CAF, B&P+ for Category (B)



[23] Dell'Amico, M., Furini, F., Iori, M.: A branch-and-price algorithm for the temporal bin packing problem. Computers & Operations Research 114, Article 104825 (2020)

TBPP-FU: comparison with [38] for Category (A1)

- compact M1-type model from [38] with many improvements w.r.t. Aydin *et al.* '2020 is the best solution approach from the literature:

n	$\bar{s}$	d_i	c_i	CAF t_{mod}	CAF t	CAF opt	[38] t	[38] opt
50	50	d_S	c_L	1.0	1.6	(5)	4.0	(5)
			c_H	0.2	0.1	(5)	1.3	(5)
		d_L	c_L	1.7	0.8	(5)	360.8	(4)
			c_H	0.5	0.1	(5)	0.5	(5)
	60	d_S	c_L	0.6	0.9	(5)	1.2	(5)
			c_H	0.3	0.2	(5)	1.7	(5)
		d_L	c_L	2.0	1.2	(5)	11.2	(5)
			c_H	0.5	0.1	(5)	0.6	(5)
Average (Sum)				0.9	**0.6**	**(40)**	47.7	(39)
100	100	d_S	c_L	2.1	4.2	(5)	3.6	(5)
			c_H	0.7	0.3	(5)	78.6	(5)
		d_L	c_L	10.9	78.2	(5)	1449.8	(1)
			c_H	3.0	1.7	(5)	1092.9	(2)
	120	d_S	c_L	1.3	3.1	(5)	85.3	(5)
			c_H	0.5	0.3	(5)	83.6	(5)
		d_L	c_L	5.6	13.7	(5)	685.9	(4)
			c_H	1.7	0.7	(5)	546.9	(4)
Average (Sum)				3.2	**12.8**	**(40)**	503.3	(31)

- t_{mod} : time to build the network and the corresponding ILP model.

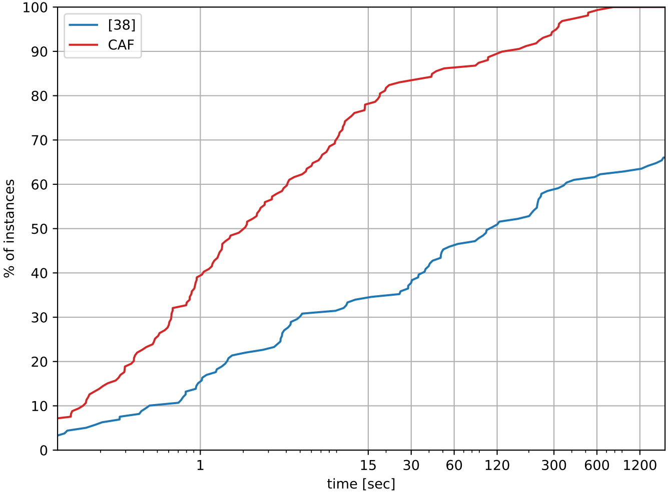
(cont.)

TBPP-FU: comparison with [38] for Category (A1) (cont.)

				CAF			[38]	
n	$\bar{s}$	d_i	c_i	t_{mod}	t	opt	t	opt
150	150	d_S	c_L	3.2	13.2	(5)	85.7	(5)
			c_H	1.0	0.6	(5)	1464.1	(1)
		d_L	c_L	18.9	177.1	(5)	1462.2	(1)
			c_H	4.2	4.2	(5)	1372.0	(2)
	180	d_S	c_L	2.1	8.0	(5)	37.2	(5)
			c_H	0.9	0.4	(5)	853.9	(3)
		d_L	c_L	13.0	175.1	(5)	1198.4	(3)
			c_H	3.0	2.2	(5)	1494.0	(1)
Average (Sum)				5.8	**47.6**	**(40)**	995.9	(21)
200	200	d_S	c_L	5.1	47.1	(5)	99.2	(5)
			c_H	1.5	1.6	(5)	1800.0	(0)
		d_L	c_L	29.9	571.0	(5)	1800.0	(0)
			c_H	7.3	9.1	(5)	1624.6	(1)
	240	d_S	c_L	3.3	14.1	(5)	123.1	(5)
			c_H	1.2	0.7	(5)	1693.9	(2)
		d_L	c_L	18.6	283.2	(5)	1201.9	(2)
			c_H	5.0	14.2	(5)	1800.0	(0)
Average (Sum)				9.0	**117.6**	**(40)**	1267.8	(15)
Total: Average (Sum)				4.7	**44.7**	**(160)**	703.7	(106)

- note: instances with d_L and c_L are the most difficult.

TBPP-FU: performance profile for Category (A1)



[38] Martinovic, J., Strasdat, N.: Theoretical insights and a new class of valid inequalities for the temporal bin packing problem with fire-ups. Preprint MATH-NM-01-2022, Technische Universität Dresden (2022) (http://www.optimization-online.org/DB_HTML/2022/02/8791.html)

TBPP: results for large instances in Category (A2)

- so far only addressed with heuristic methods (Aydin *et al.* '2020):

					$t_{max} = 1800s$		$t_{max} = \infty$	
n	$\bar{s}$	d_i	c_i	t_{mod}	t	opt	t	opt
500	500	d_S	c_L	11.9	121.7	(5)	121.7	(5)
			c_H	4.6	4.7	(5)	4.7	(5)
		d_L	c_L	75.4	1696.3	(1)	5103.3	(5)
			c_H	19.6	144.2	(5)	144.2	(5)
	600	d_S	c_L	7.8	47.4	(5)	47.4	(5)
			c_H	2.6	2.3	(5)	2.3	(5)
		d_L	c_L	53.8	1467.3	(2)	2011.9	(5)
			c_H	13.8	47.0	(5)	47.0	(5)
Average (Sum)				23.7	441.4	(33)	935.3	(40)
1000	1000	d_S	c_L	24.5	887.5	(4)	933.4	(5)
			c_H	7.2	12.4	(5)	12.4	(5)
		d_L	c_L	150.0	1800.0	(0)	20483.1	(5)
			c_H	38.9	176.8	(5)	176.8	(5)
	1200	d_S	c_L	18.2	340.1	(5)	340.1	(5)
			c_H	5.5	8.3	(5)	8.3	(5)
		d_L	c_L	103.1	1800.0	(0)	5615.1	(5)
			c_H	28.1	126.6	(5)	126.6	(5)
Average (Sum)				46.9	644.0	(29)	3462.0	(40)
Total: Average (Sum)				35.3	542.7	(62)	2198.6	(80)

Conclusions

- Combinatorial Arc Flow (CAF) formulations are strong and the new state space definition, with representative states, significantly reduces the number of nodes and arcs.
- All benchmark TBPP and TBPP-FU instances (used in exact approaches) were solved with a significant speed-up factor.
- Optimal solutions were found for large TBPP instances only addressed previously with heuristics.
- Same concepts can be prospectively applied to other classes of optimization problems in the field of interval scheduling.

End

Thank you for your attention.

Acknowledgement

- This work has been supported by FCT - Fundação para a Ciência e Tecnologia within the R&D Units Project Scope UIDB/00319/2020.