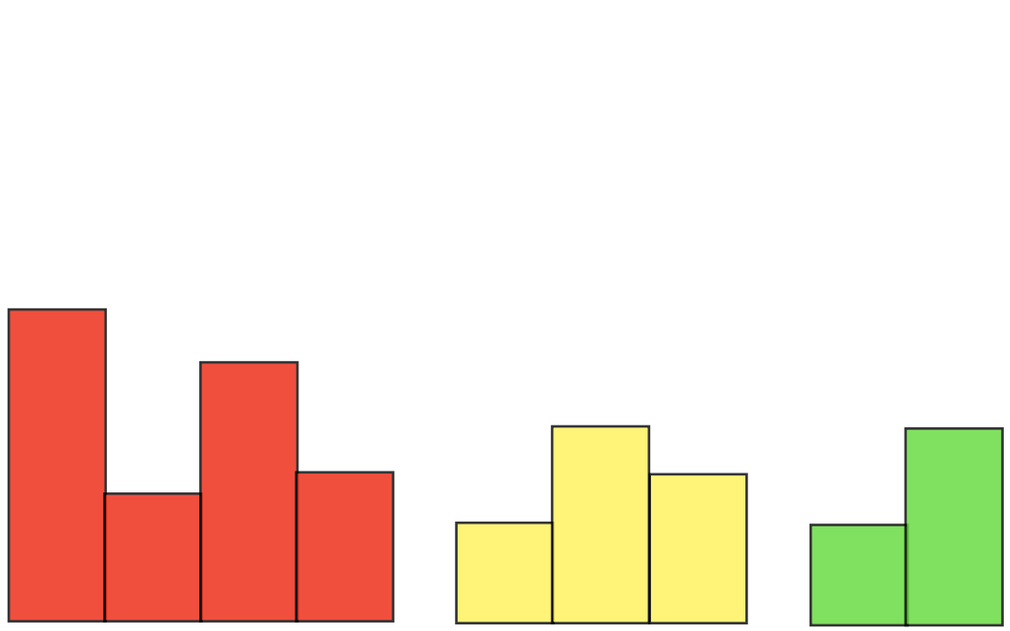
Approximation Algorithms for Two-Bar Charts Packing Problem

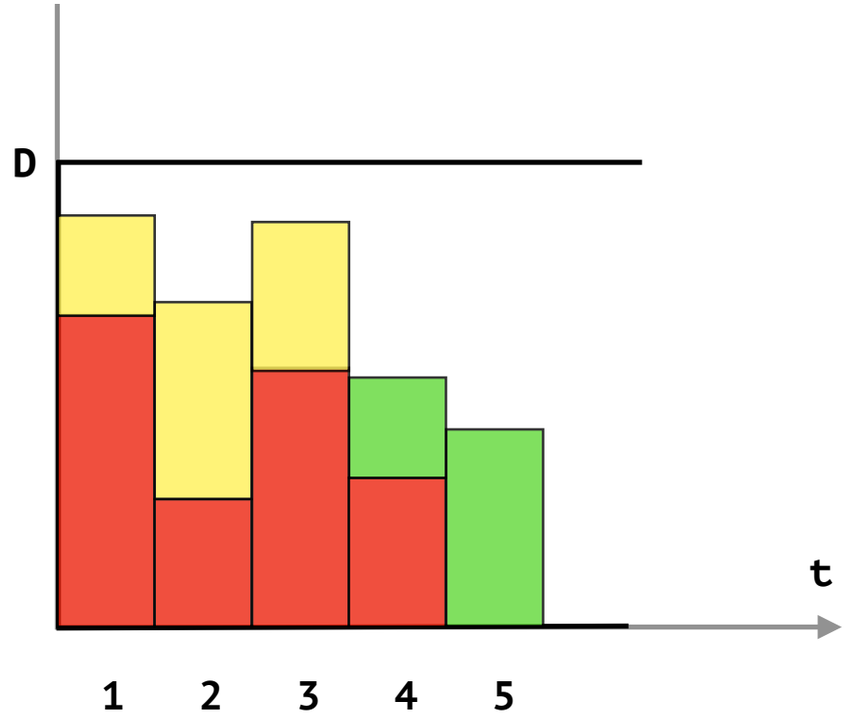
29.09.2021
OPTIMA conference

Adil Erzin, **Georgii Melidi**, Stepan Nazarenko, and Roman Plotnikov

Bar Charts Packing Problem (BCPP)

In **[1]** we studied the problem of optimizing an investment portfolio in the oil and gas field. Each project is characterized by the annual volume of hydrocarbon production, adequately displayed with bar charts. Each year, the total production volume of all projects must not exceed a given value (throughput of a pipe). The problem is to complete all projects as early as possible.

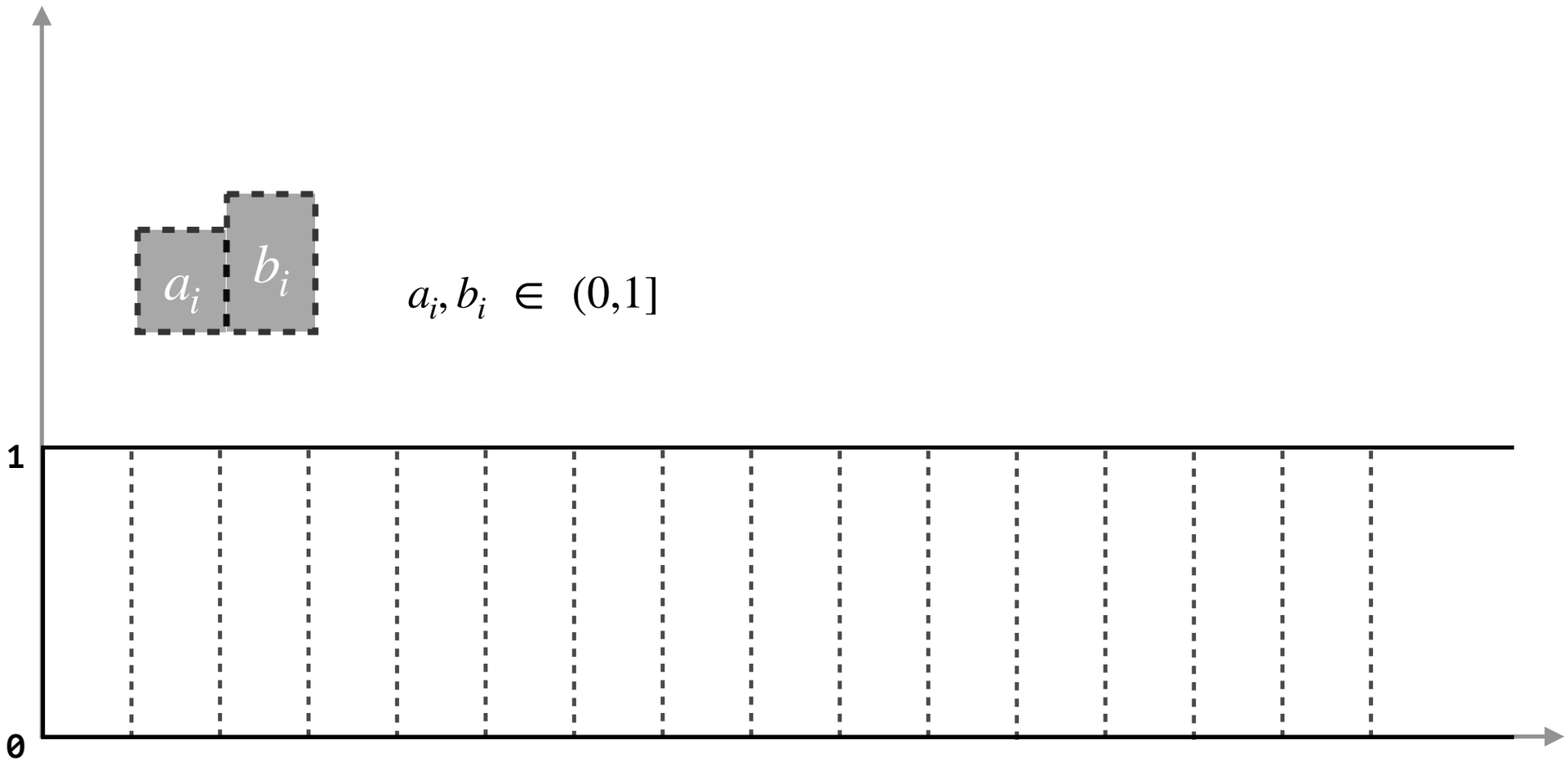




[1] **Erzin A., et al.: Optimal Investment in the Development of Oil and Gas Field. Mathematical Optimization Theory and Operations Research. MOTOR 2020. CCIS, vol 1275, 336-349. Springer, Cham. (2020)**

Two-Bar Charts Packing Problem (2-BCPP)

The 2-BCPP was first formulated in **[1]** and then examined in **[2-4]**.



[1] **Erzin A., et al.: Optimal Investment in the Development of Oil and Gas Field. Mathematical Optimization Theory and Operations Research. MOTOR 2020. CCIS, vol 1275, 336-349. Springer, Cham. (2020)**
[2] **Erzin A., et al.: Two-Bar Charts Packing Problem. Optimization Letters. https://doi.org/10.1007/s11590-020-01657-1 (2020)**
[3] **Erzin A.,et al.:A 3/2-approximation for big two-bar charts packing.J.of Combinatorial Optimization. https://doi.org/10.1007/s10878-021-00741-1 (2021)**
[4] **Erzin A.,et al.: A Posteriori Analysis of the Algorithms for Two-Bar Charts Packing Problem. Communications in Computer and Information Science. URL: https://doi.org/10.1007/978-3-030-92711-0_14 (2021)**

Formulation of the problem

$$x_{ij} = \begin{cases} 1, & \text{if the first bar of BC } i \text{ is in the cell } j; \\ 0, & \text{else.} \end{cases}$$

$$y_j = \begin{cases} 1, & \text{if the cell } j \text{ contains at least one bar;} \\ 0, & \text{else.} \end{cases}$$

$$\sum_j y_j \to \min_{x_{ij}, y_j \in \{0,1\}}; \qquad (1)$$

$$\sum_j x_{ij} = 1, \quad i \in S; \qquad (2)$$

$$\sum_i a_i x_{ij} + \sum_k b_k x_{k,j-1} \le y_j, \quad \forall j. \qquad (3)$$

Similar problems

Bin Packing Problem

- FFD(L) ⩽ 11/9 OPT(L) + 4 (Johnson - 1973)
- FFD(L) ⩽ 11/9 OPT(L) + 3 (Backer - 1985)
- FFD(L) ⩽ 11/9 OPT(L) + 1 (Yue - 1991)
- FFD(L) ⩽ 11/9 OPT(L) + 7/9 (Li, Yue - 1997)
- FFD(L) ⩽ 11/9 OPT(L) + 6/9 (Dósa - 2007)
- MFFD(L) ⩽ 71/60 OPT(L) + 31/6 (Johnson, Garey - 1985)
- MFFD(L) ⩽ 71/60 OPT(L) + 1 (Yue, Zhang - 1995)

2D Vector Packing Problem

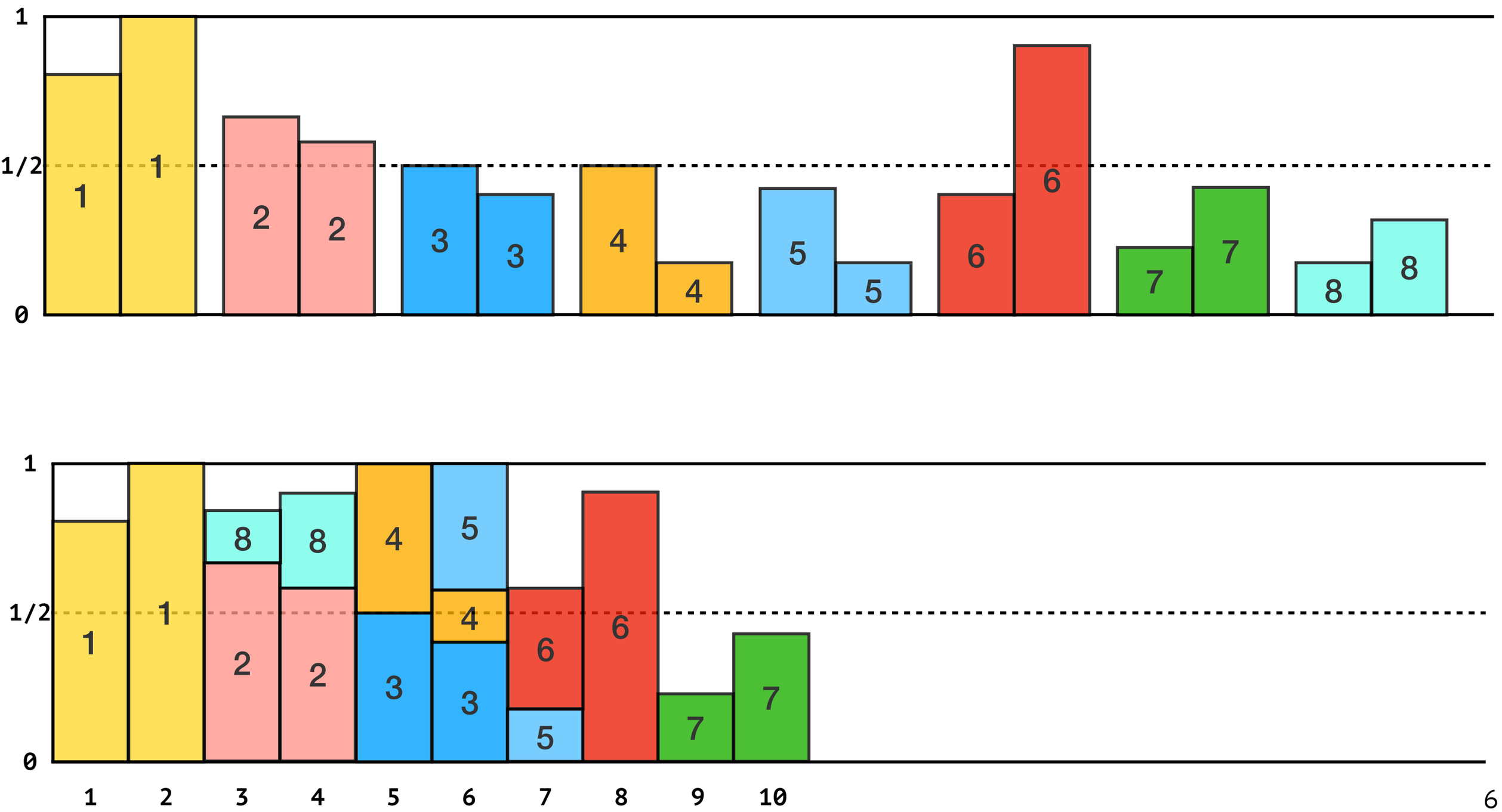
- W(L) ⩽ 2 OPT(L) (Kellerer, Kotov - 2003)
- W(L) ⩽ (3/2+ε) OPT(L) (Bansal et al. - 2016)

Strip Packing Problem

- W(L) ⩽ 3 OPT(L) (Baker - 1980)
- W(L) ⩽ 2.7 OPT(L) (Coffman et al. - 1980)
- W(L) ⩽ 2.5 OPT(L) (Sleator - 1980)
- W(L) ⩽ 2 OPT(L) (Schiermeyer - 1994, Steinberg - 1997)
- W(L) ⩽ (5/3+ε) OPT(L) (Harren et al. - 2014)

Algorithms

Greedy algorithm GA_LO



Algorithms

Greedy algorithm GA_LO

In **[2]**, we proved the following

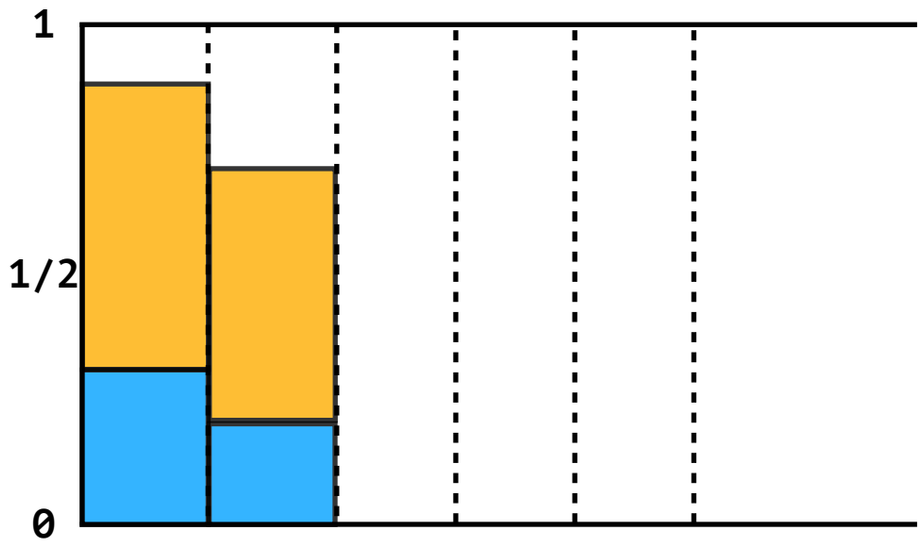
Theorem 1. Algorithm A with time complexity $O(n^2)$ constructs a packing for 2-BCPP whose length is at most 2OPT +1, where OPT is the minimum length of a strip into which n 2-BCs can be packed.

[2] **Erzin A., et al.: Two-Bar Charts Packing Problem. Optimization Letters. https://doi.org/10.1007/s11590-020-01657-1 (2020)**

Algorithms

Algorithms M_w and M1_w

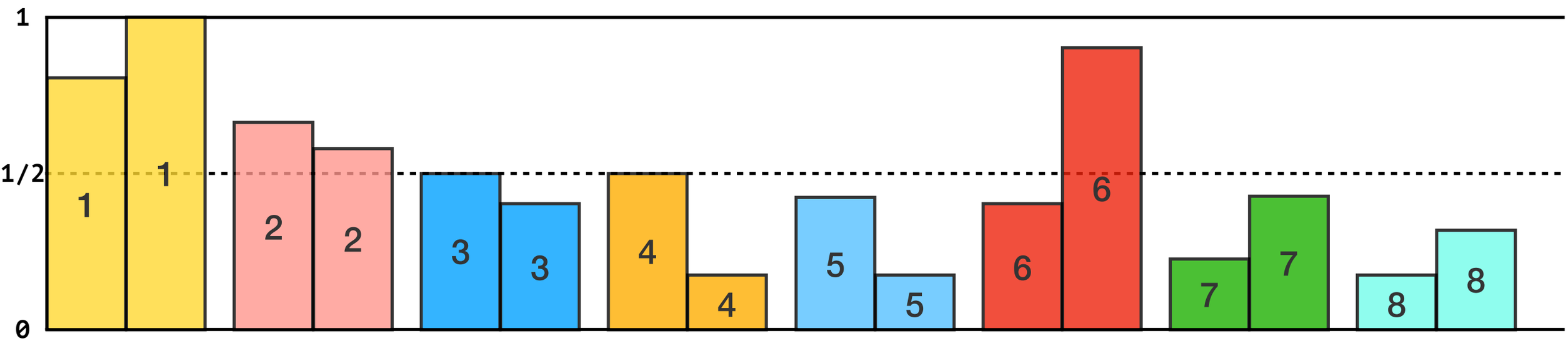
Definition 1. Two arbitrary BCs create a t-union if they are packed so that t cells of the strip contain the bars of both BCs.

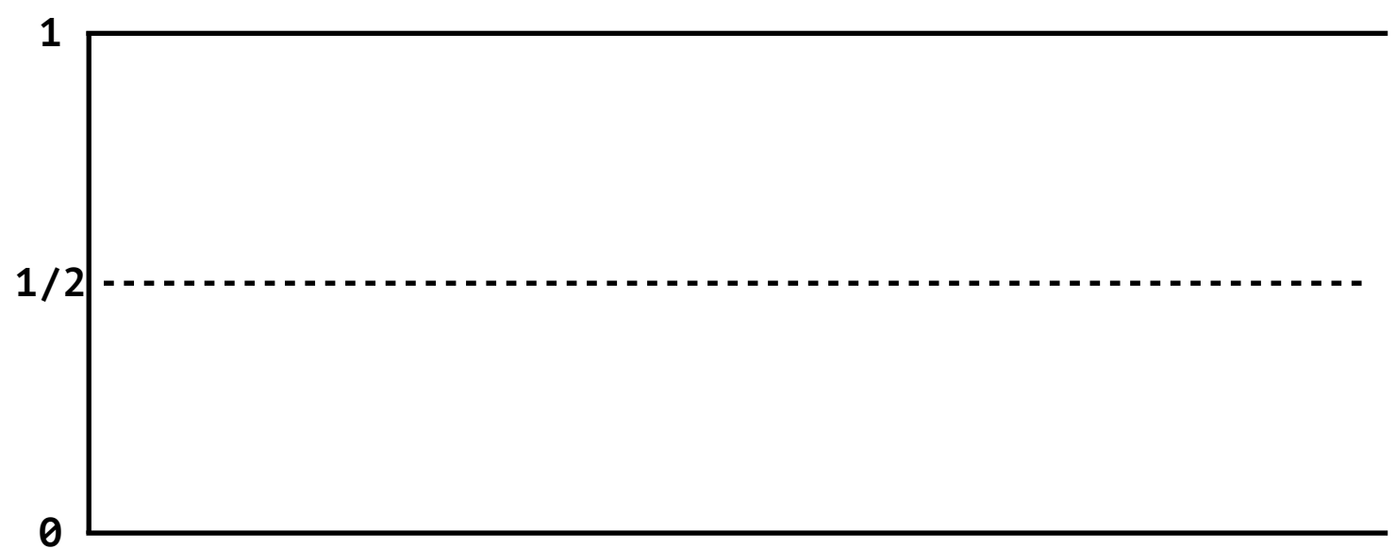


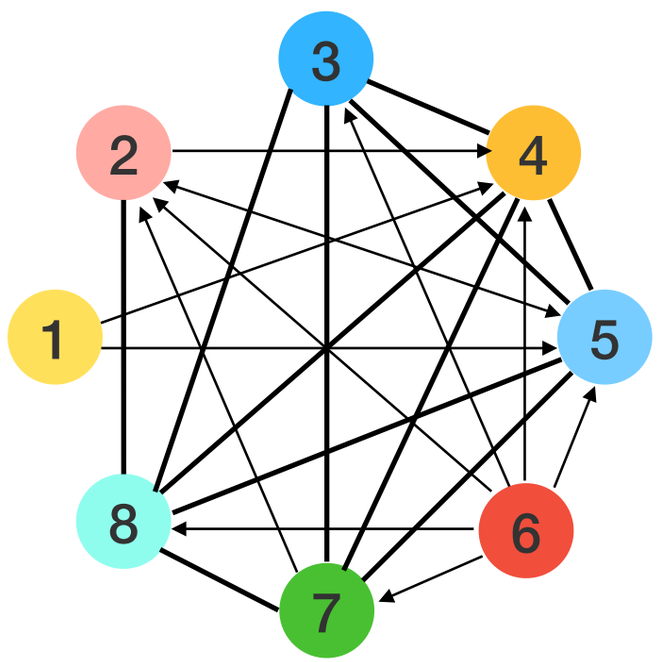
2-union

Algorithms

Algorithms M_w and M1_w

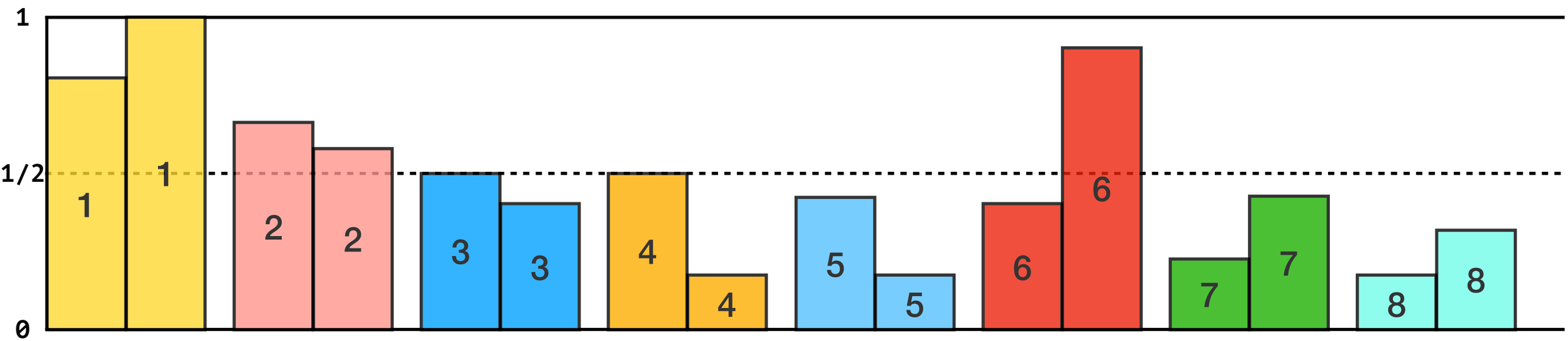


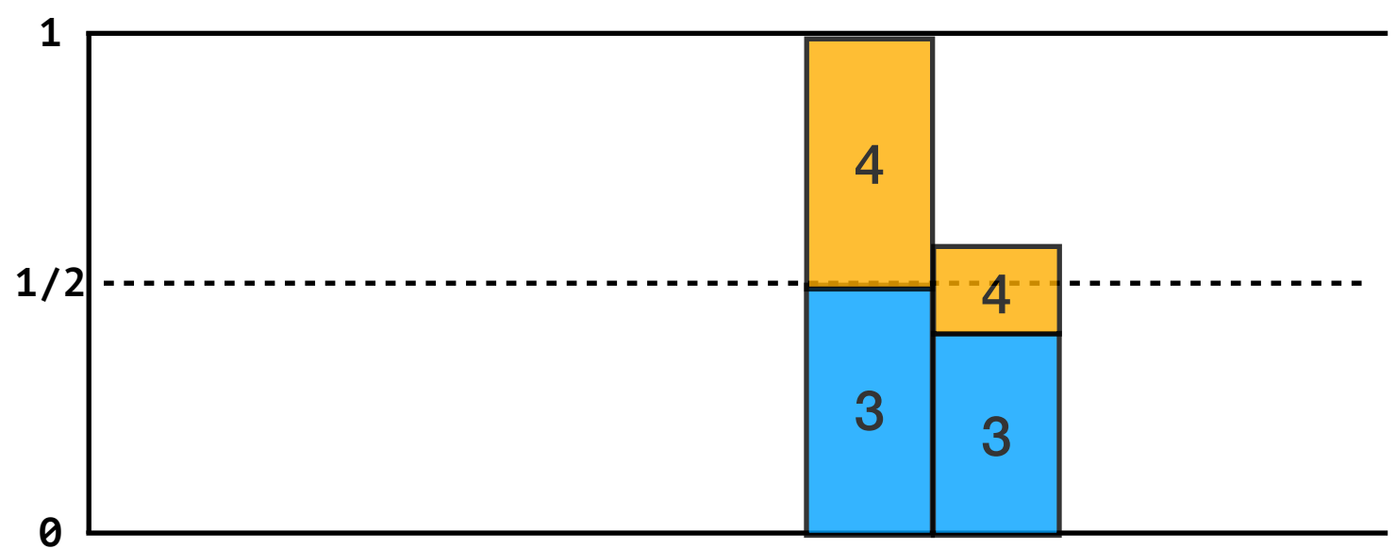


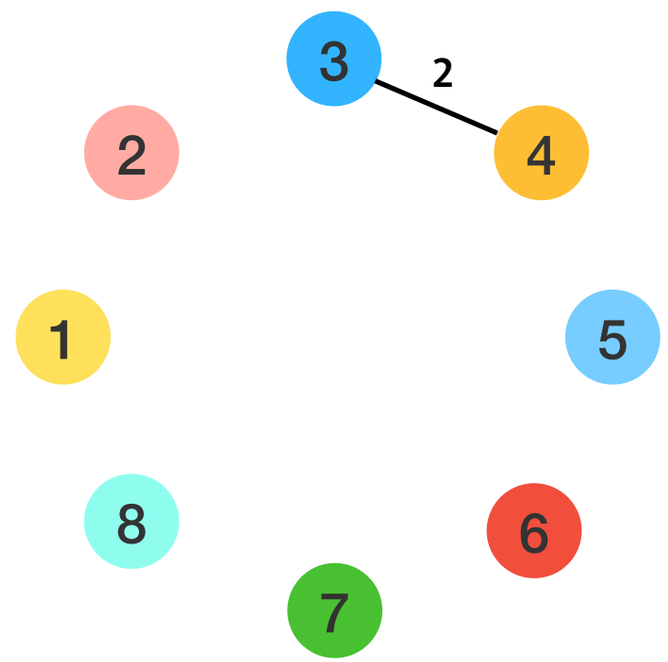


Algorithms

Algorithms M_w and M1_w

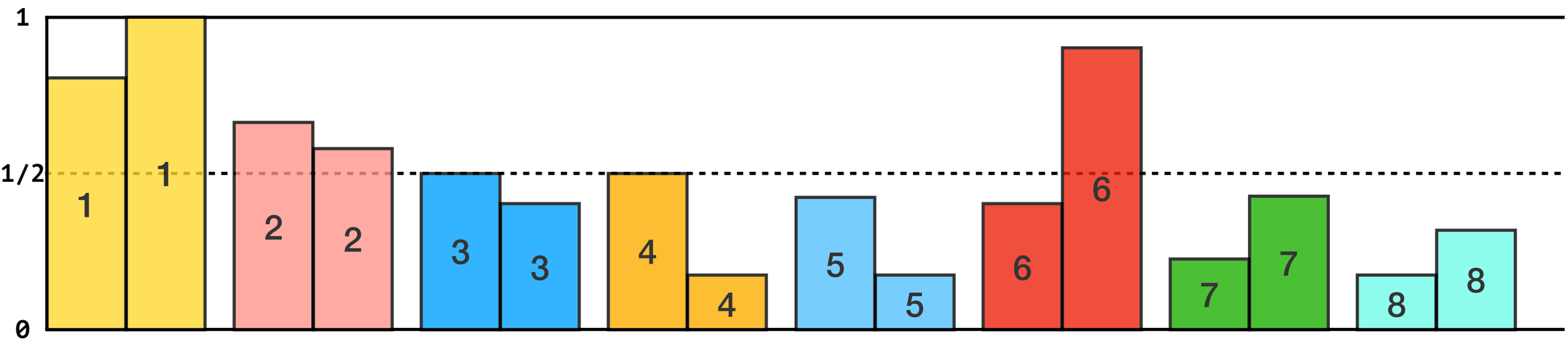


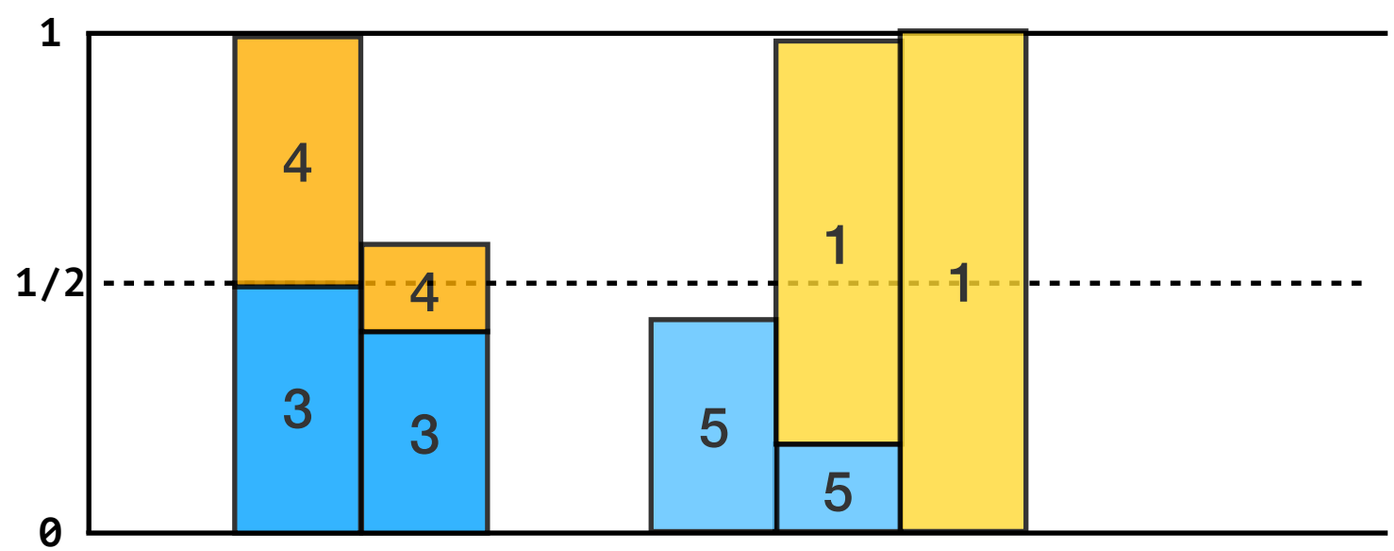


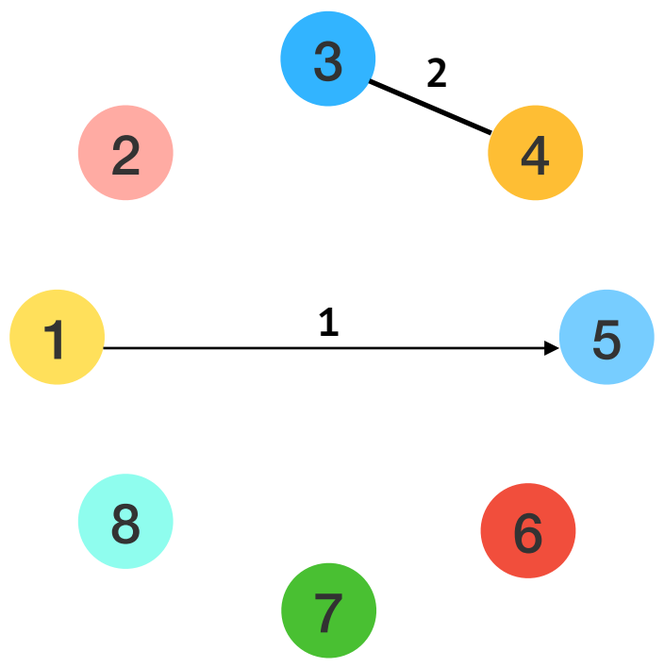


Algorithms

Algorithms M_w and M1_w

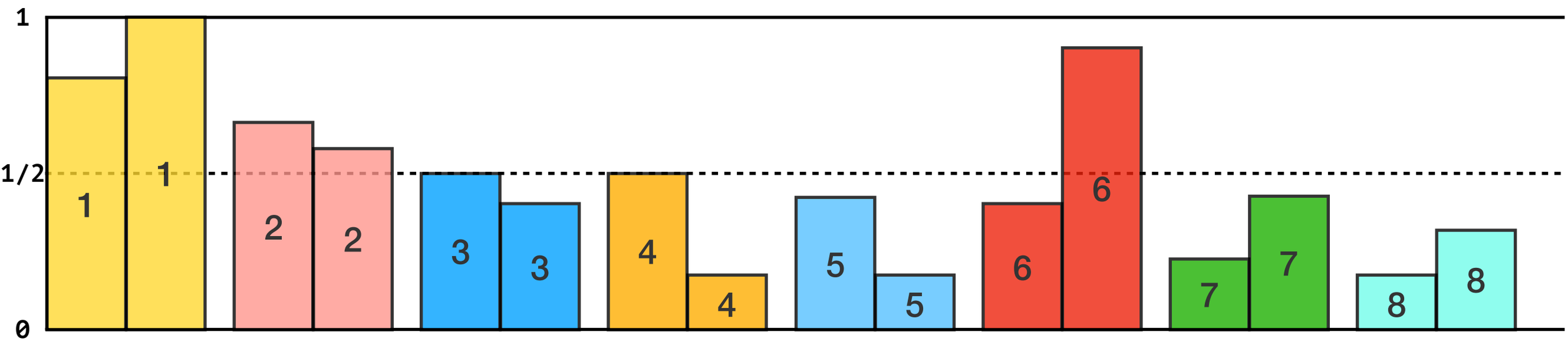


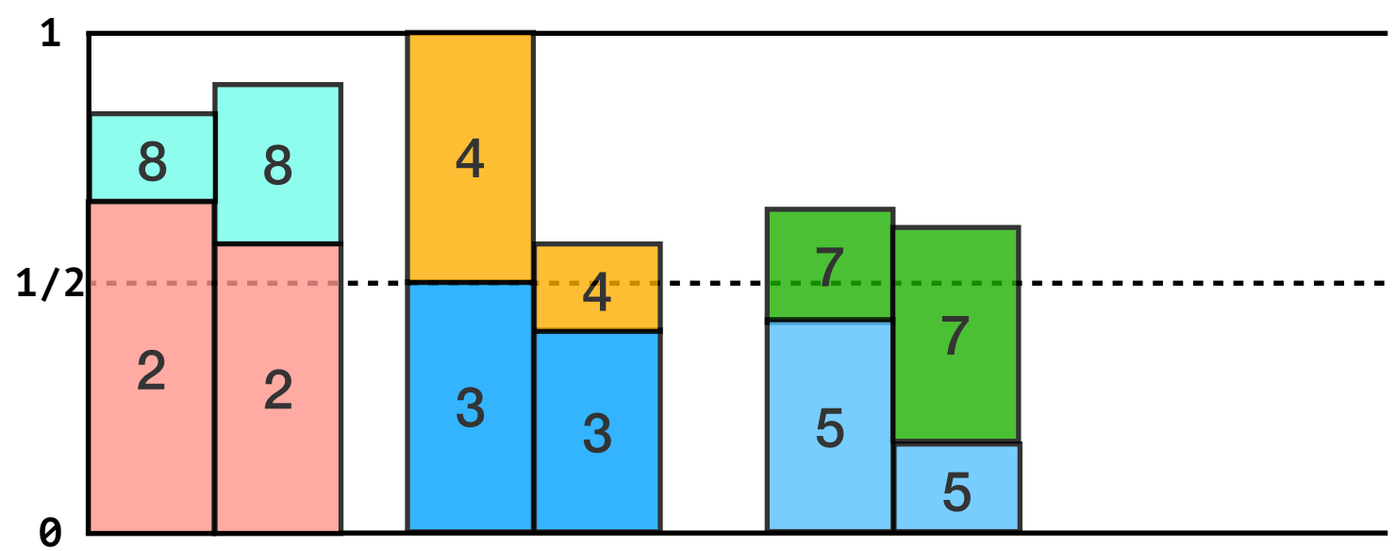


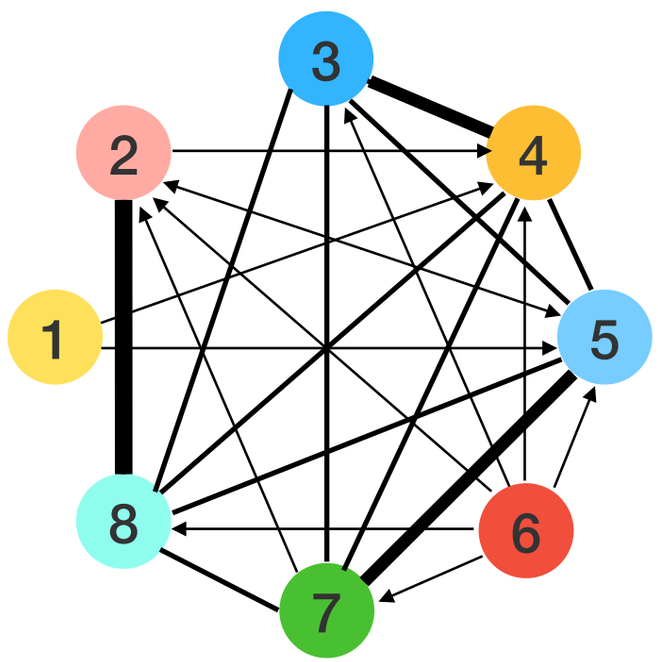


Algorithms

Algorithms M_w and M1_w

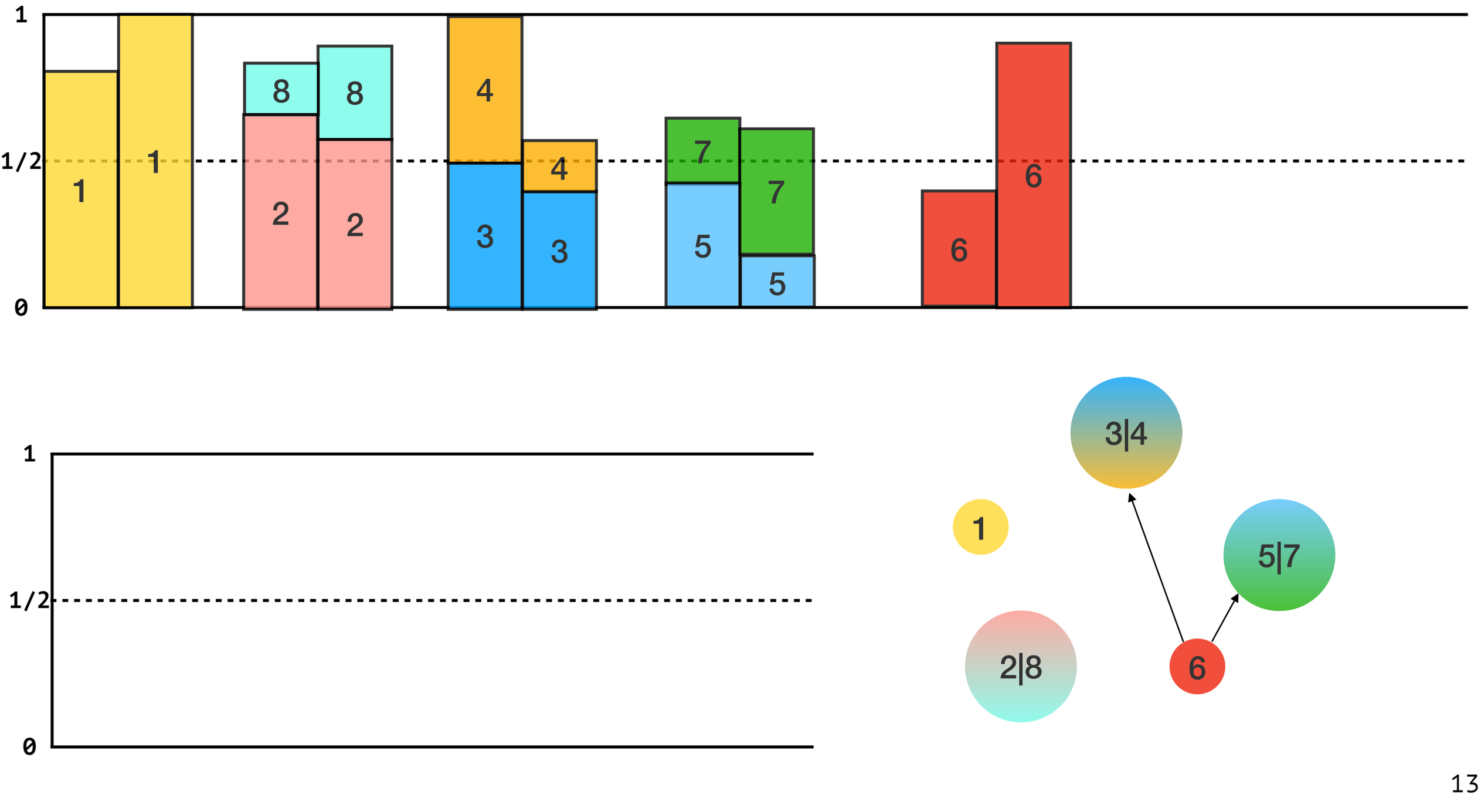






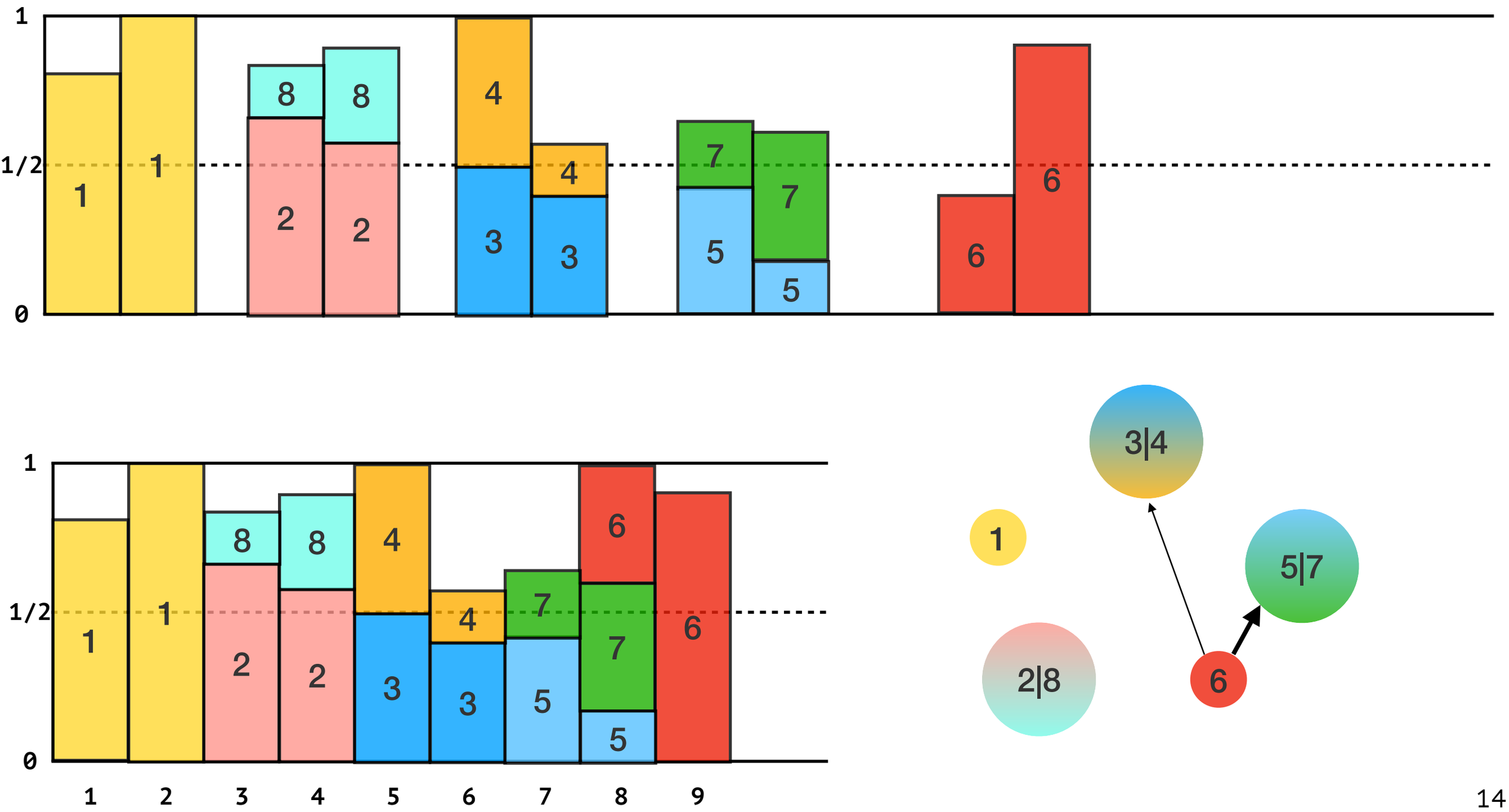
Algorithms

Algorithms M_w and M1_w



Algorithms

Algorithms **M_w** and M1_w



Algorithms

Algorithms M_w and M1_w

Theorem 2. **[3]** The time complexity of algorithm M_w is **$O(n^4)$**, and if all 2-BCs are big, it constructs a **3/2-approximate solution** to the 2-BCPP.

From the proof of this theorem [3] follows the

Remark 1. In order to achieve the corresponding accuracy, it is sufficient to construct only the first max-weight matching. Thus, the complexity of obtaining a **3/2-approximate solution is $O(n^3)$**.

Definition 1. Two arbitrary BCs create a t-union if they are packed so that t cells of the strip contain the bars of both BCs.

Proposition 1. The algorithm M_w builds only 1- and 2-unions.

[3] **Erzin A.,et al.:A 3/2-approximation for big two-bar charts packing.J.of Combinatorial Optimization. https://doi.org/10.1007/s10878-021-00741-1 (2021)**

Algorithms

Algorithms A1 and A2

A1 and **A2** are based on the reduction of 2-BCPP to MaxATSP(0,1). The only difference is in their first parts during construction big 2-BCs.

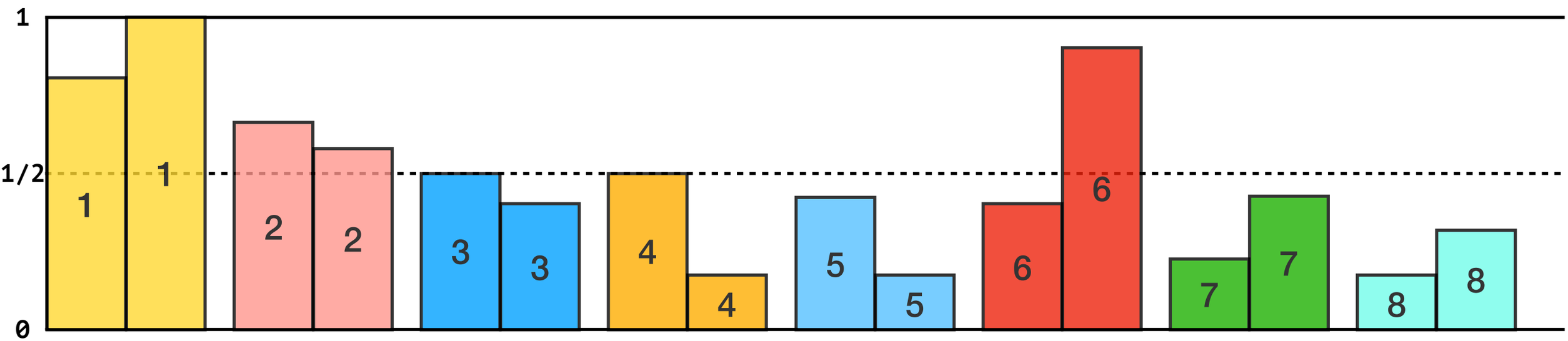
In **[4]** an **O(n³)-time 16/11-approximation algorithm** for the packing big 2-BCs is presented. To obtain this estimate, the algorithm for **construction a max-cardinality matching [5]**, and **approximation algorithm** proposed in **[6]** is used.

If 2-BCs are arbitrary, then we need to construct big 2-BCs to apply the considered algorithm. Therefore, we propose two different procedures for constructing big 2-BCs followed by applying the common part and call these algorithms **A1 and A2**.

[4] **Erzin A., Shenmaier V.: An Improved Approximation for Packing Big Two-Bar Charts. http://arxiv.org/abs/2101.00470 (2021)**
[5] **Gabow H.: An efficient reduction technique for degree-constrained subgraph and bidirected network flow problems. STOC, 448-456 (1983)**
[6] **Paluch K.: Maximum ATSP with Weights Zero and One via Half-Edges. Theory Comput. Syst. 62(2), 319-336 (2018)**

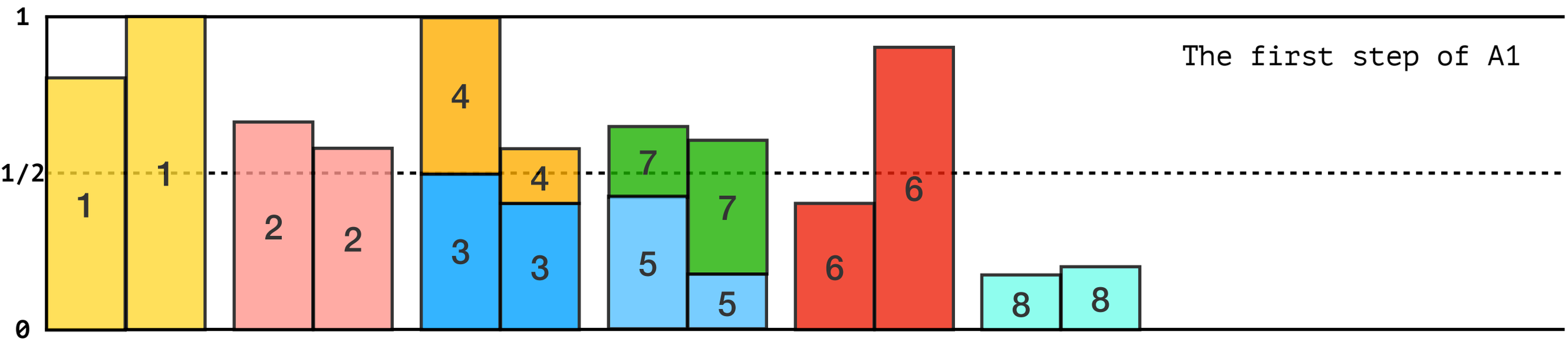
Algorithms

Algorithms **A1** and A2



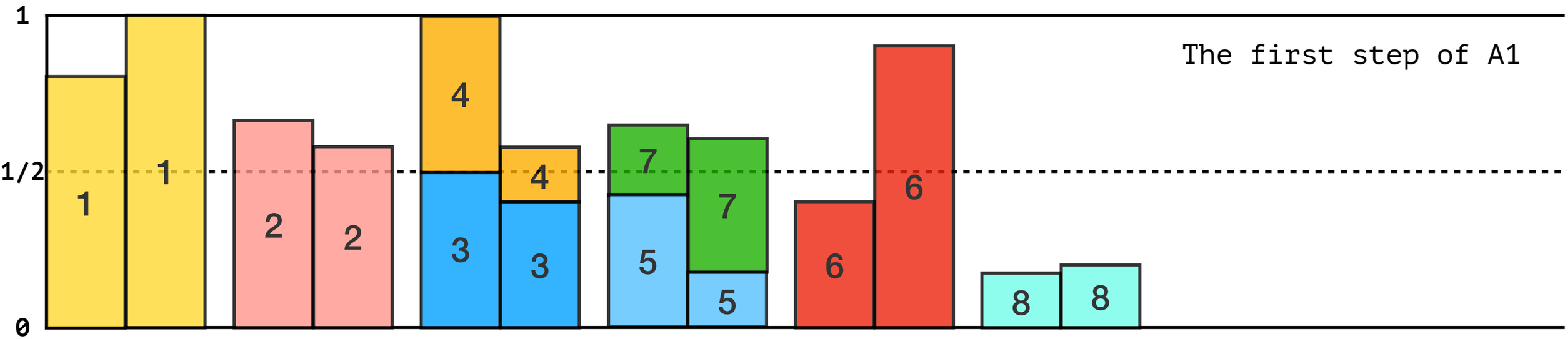
Algorithms

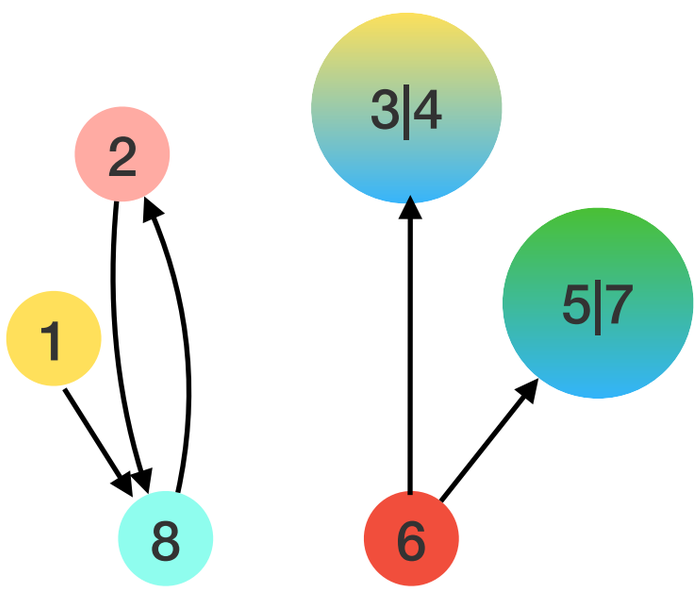
Algorithms **A1** and A2

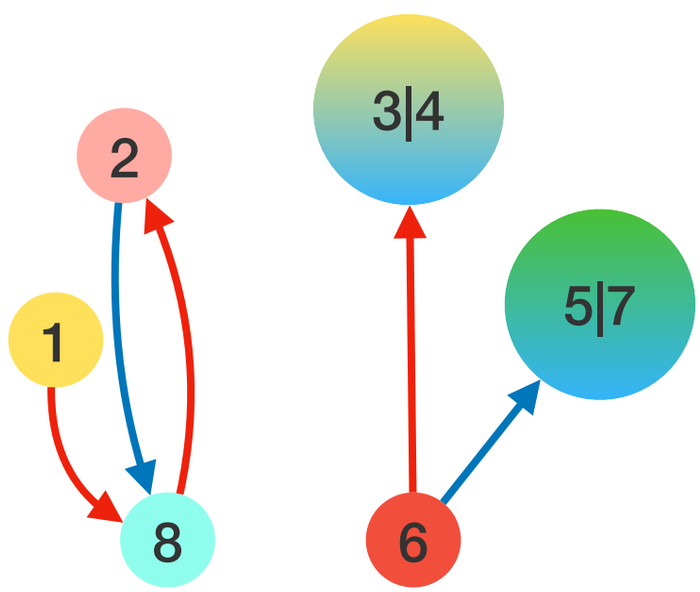


Algorithms

Algorithms **A1** and A2

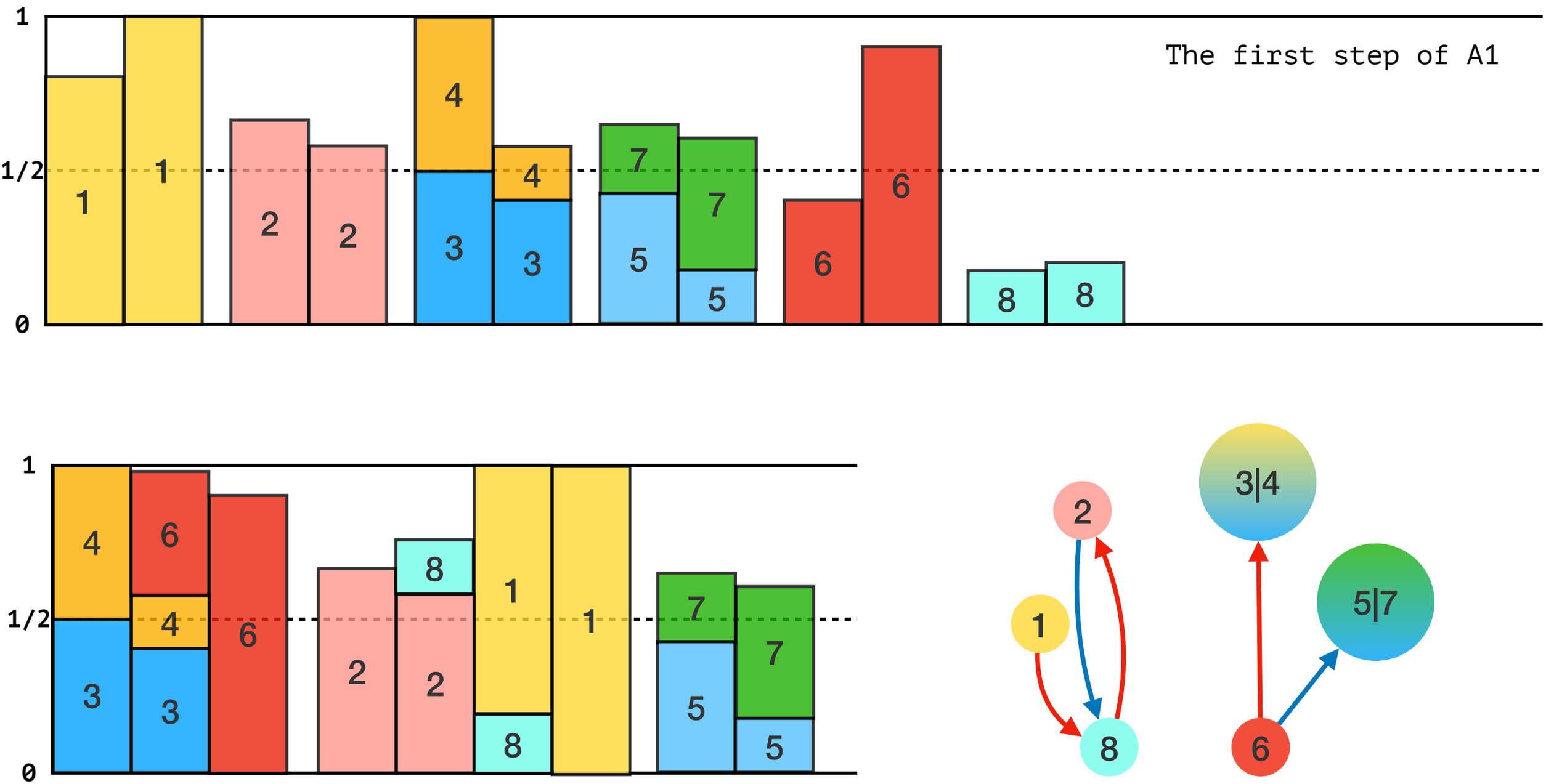






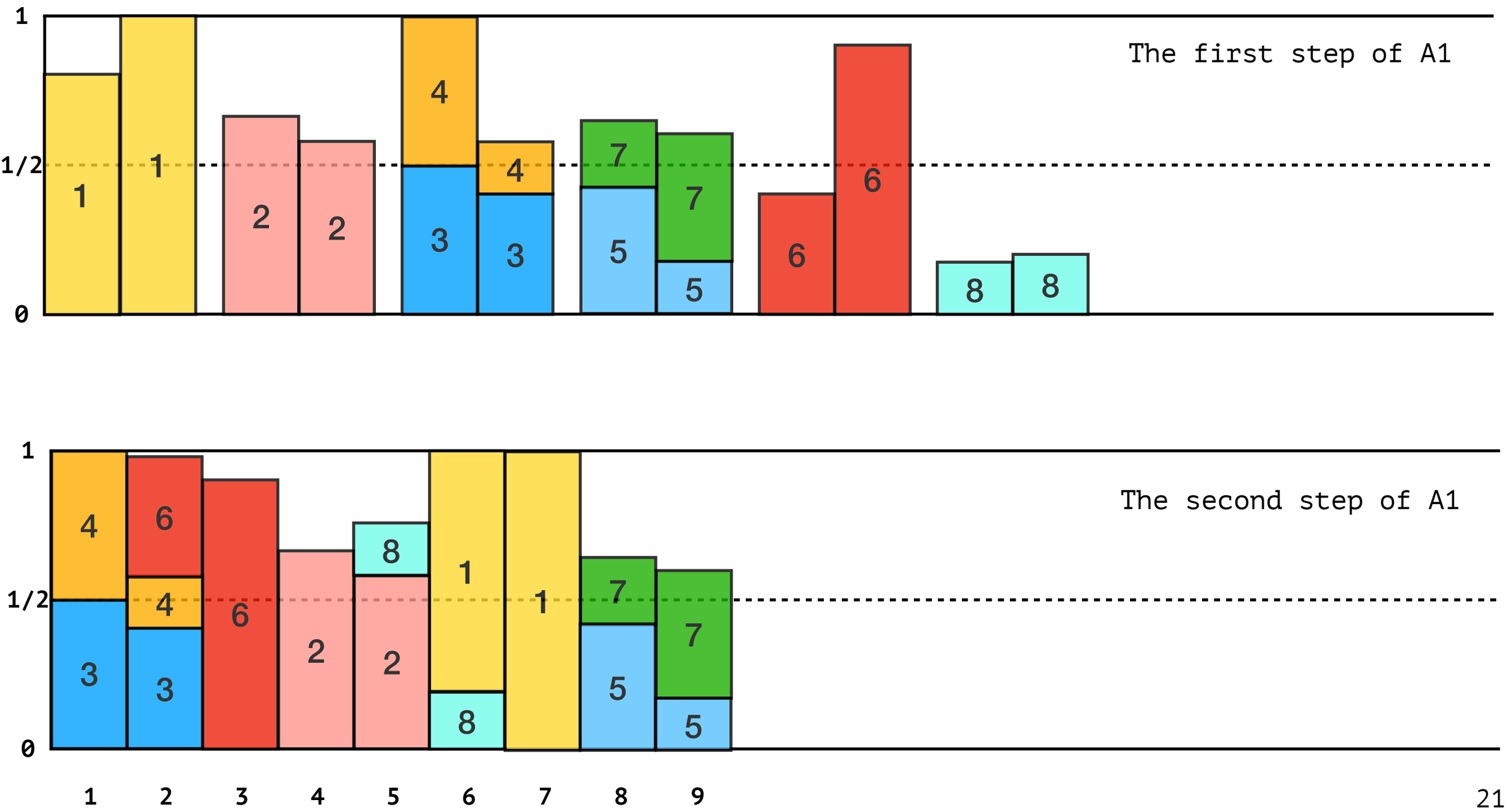
Algorithms

Algorithms **A1** and A2



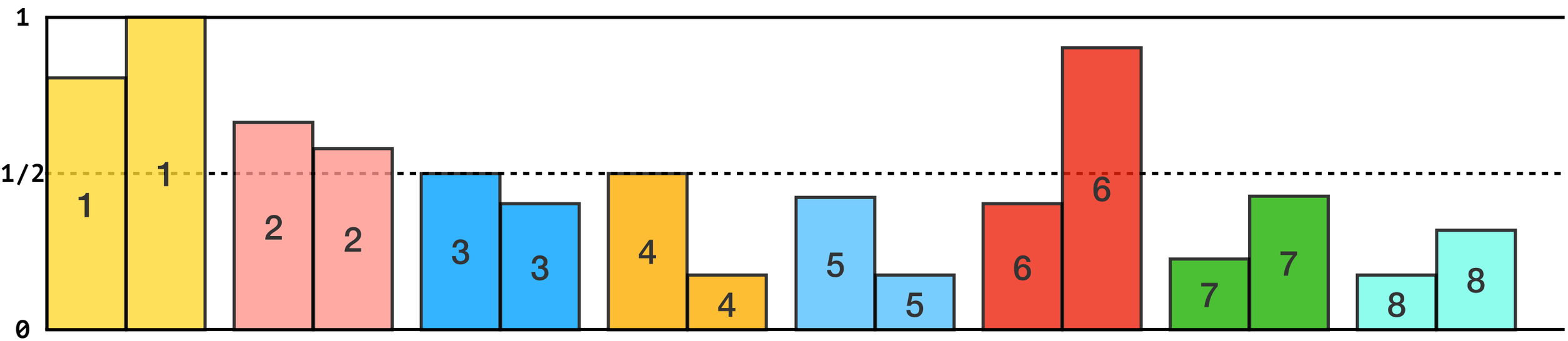
Algorithms

Algorithms **A1** and A2



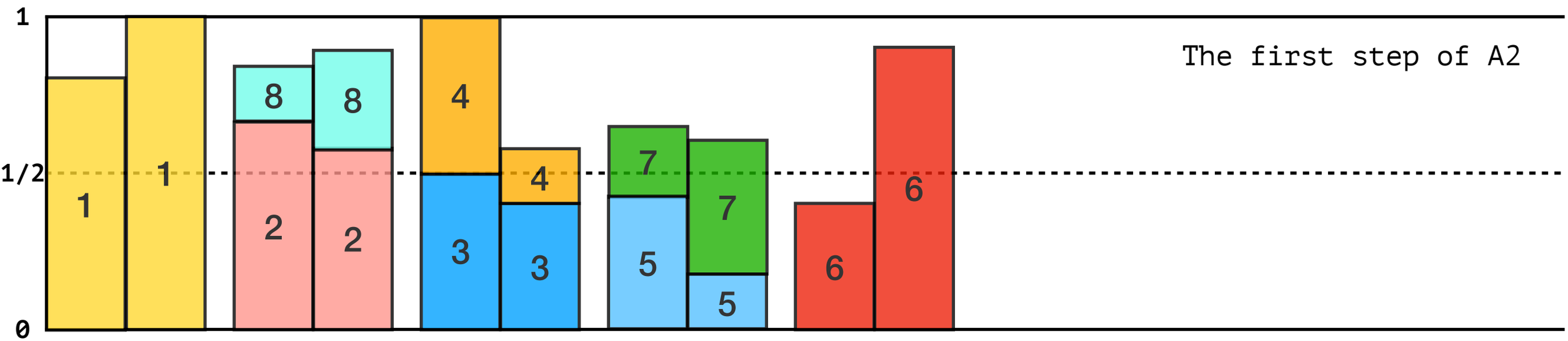
Algorithms

Algorithms A1 and **A2**



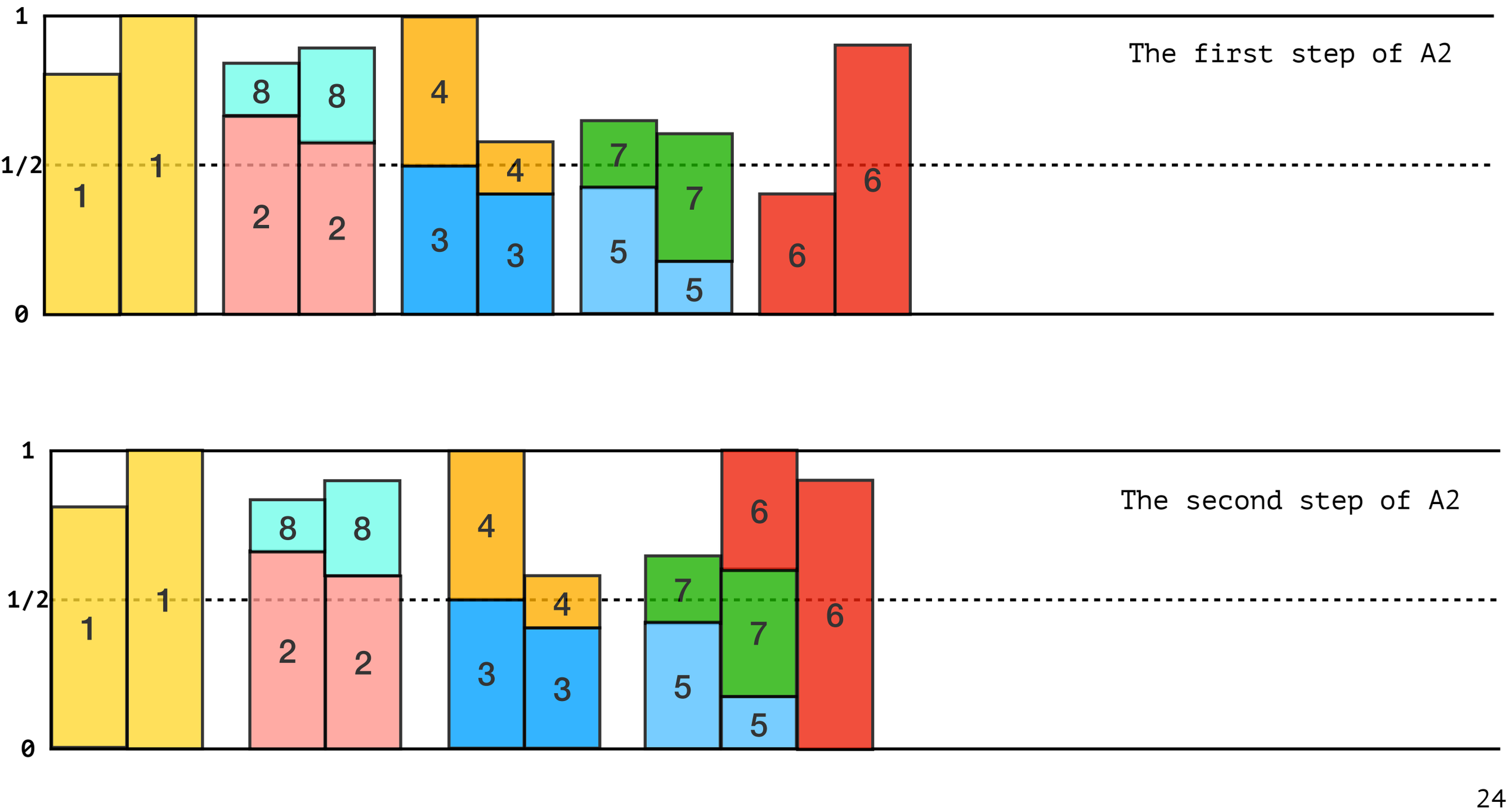
Algorithms

Algorithms A1 and **A2**



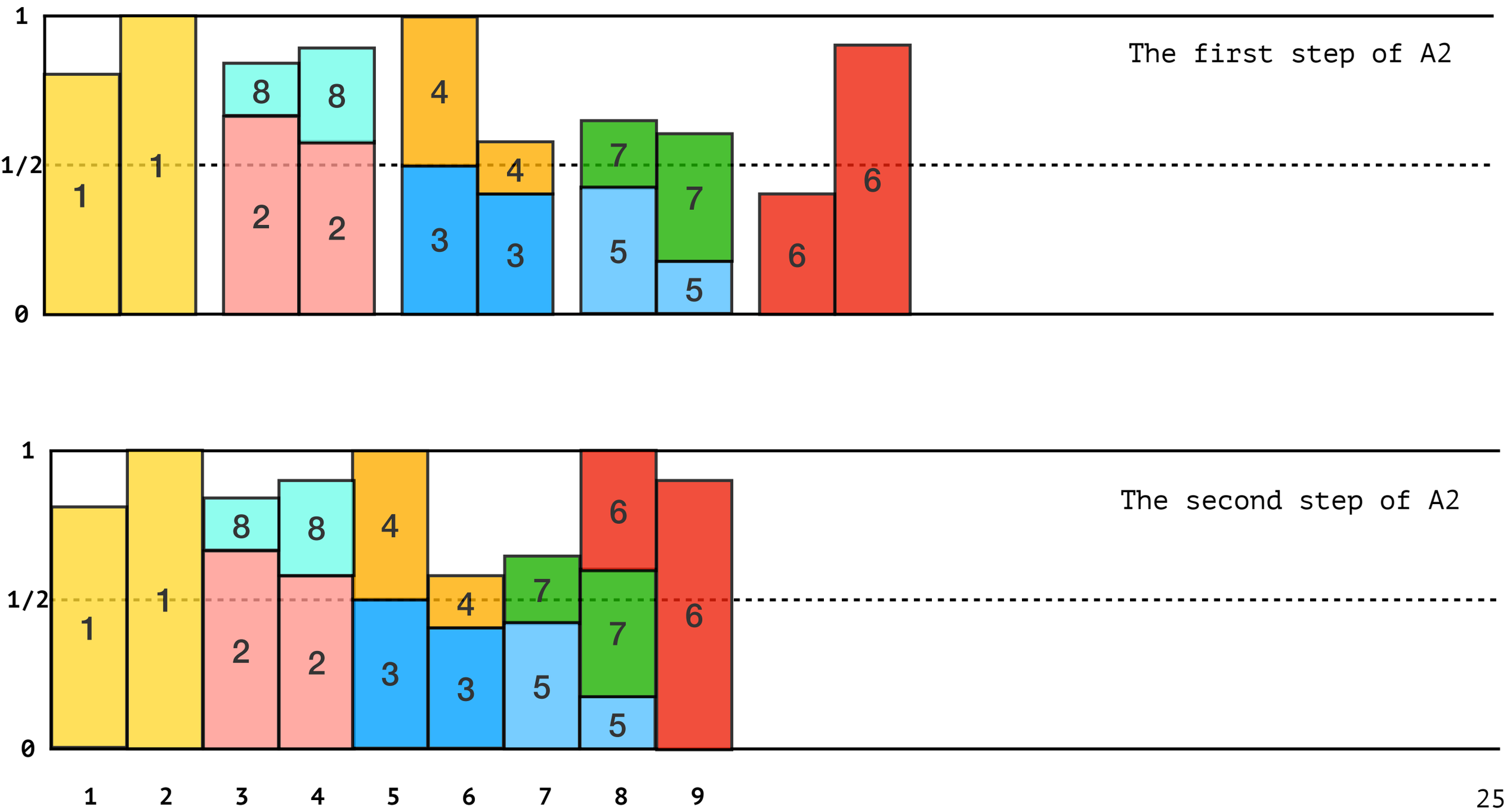
Algorithms

Algorithms A1 and **A2**



Algorithms

Algorithms A1 and **A2**



Simulation

Simulation was performed on two groups of test data.

The first one consists of **randomly generated data**.

For the generation of the instances of the second group, we used the existing collection of the **BPP instances with known optimal solutions**.

The considered algorithms have been implemented in the Python programming language. The calculations are carried out on the computer Intel Core i7-3770 3.40GHz 16Gb RAM.

Randomly generated data

For each n, ($n \in [25,1000]$) **50 different instances** are generated. To build an optimal solution to BLP or to find a lower bound for optimum, we use the IBM ILOG **CPLEX** software package with a limited running time of **300 seconds**.

We examine the approximation algorithms A1, A2, M1_W, M_W and GA_LO using three test data sets:

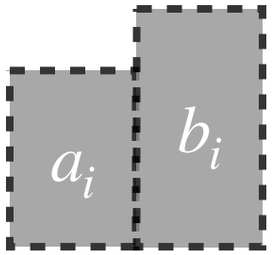
- Arbitrary 2-BCs
- Big 2-BCs
- Big non-increasing 2-BCs

Randomly generated data

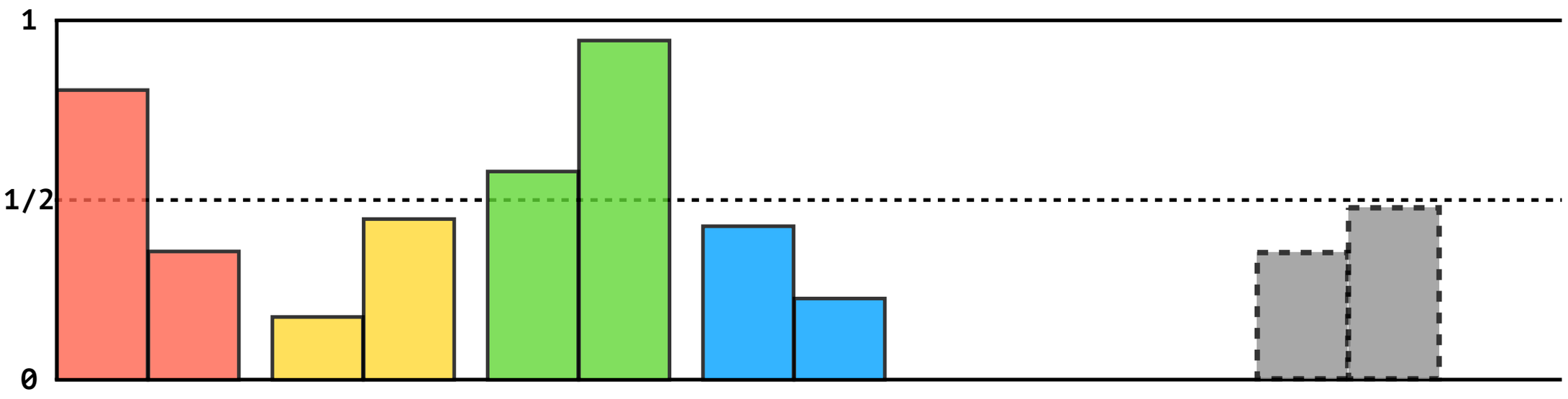
For each n, ($n \in [25,1000]$) **50 different instances** are generated. To build an optimal solution to BLP or to find a lower bound for optimum, we use the IBM ILOG **CPLEX** software package with a limited running time of **300 seconds**.

We examine the approximation algorithms A1, A2, M1_W, M_W and GA_LO using three test data sets:

- **Arbitrary 2-BCs**
- Big 2-BCs
- Big non-increasing 2-BCs



a_i, b_i **take random uniformly distributed values in** $(0,1]$

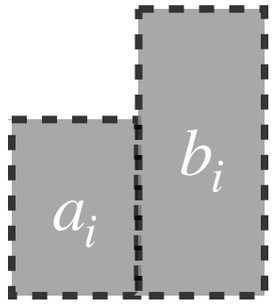


Randomly generated data

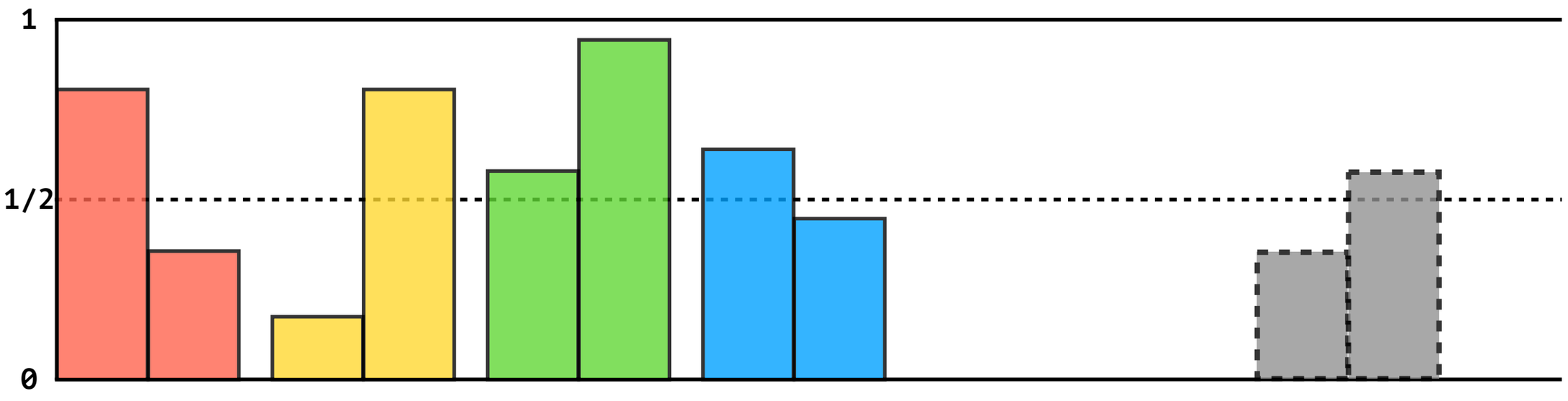
For each n, ($n \in [25,1000]$) **50 different instances** are generated. To build an optimal solution to BLP or to find a lower bound for optimum, we use the IBM ILOG **CPLEX** software package with a limited running time of **300 seconds**.

We examine the approximation algorithms A1, A2, M1_W, M_W and GA_LO using three test data sets:

- Arbitrary 2-BCs
- **Big 2-BCs**
- Big non-increasing 2-BCs



a_i or b_i take random uniformly distributed values in $(0.5,1]$

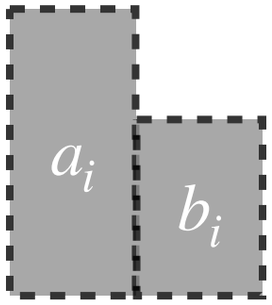


Randomly generated data

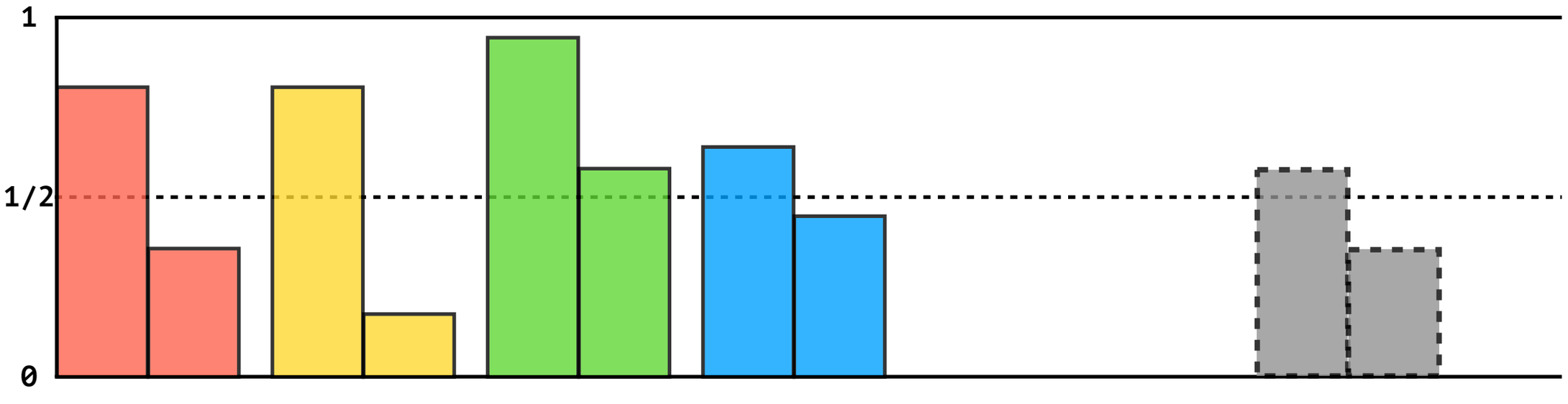
For each n, ($n \in [25,1000]$) **50 different instances** are generated. To build an optimal solution to BLP or to find a lower bound for optimum, we use the IBM ILOG **CPLEX** software package with a limited running time of **300 seconds**.

We examine the approximation algorithms A1, A2, M1_W, M_W and GA_LO using three test data sets:

- Arbitrary 2-BCs
- Big 2-BCs
- **Big non-increasing 2-BCs**



a_i or b_i take random uniformly distributed values in $(0.5,1]$
$a_i \geq b_i$



Randomly generated data

Results of the numerical experiment
with the arbitrary 2-BCs

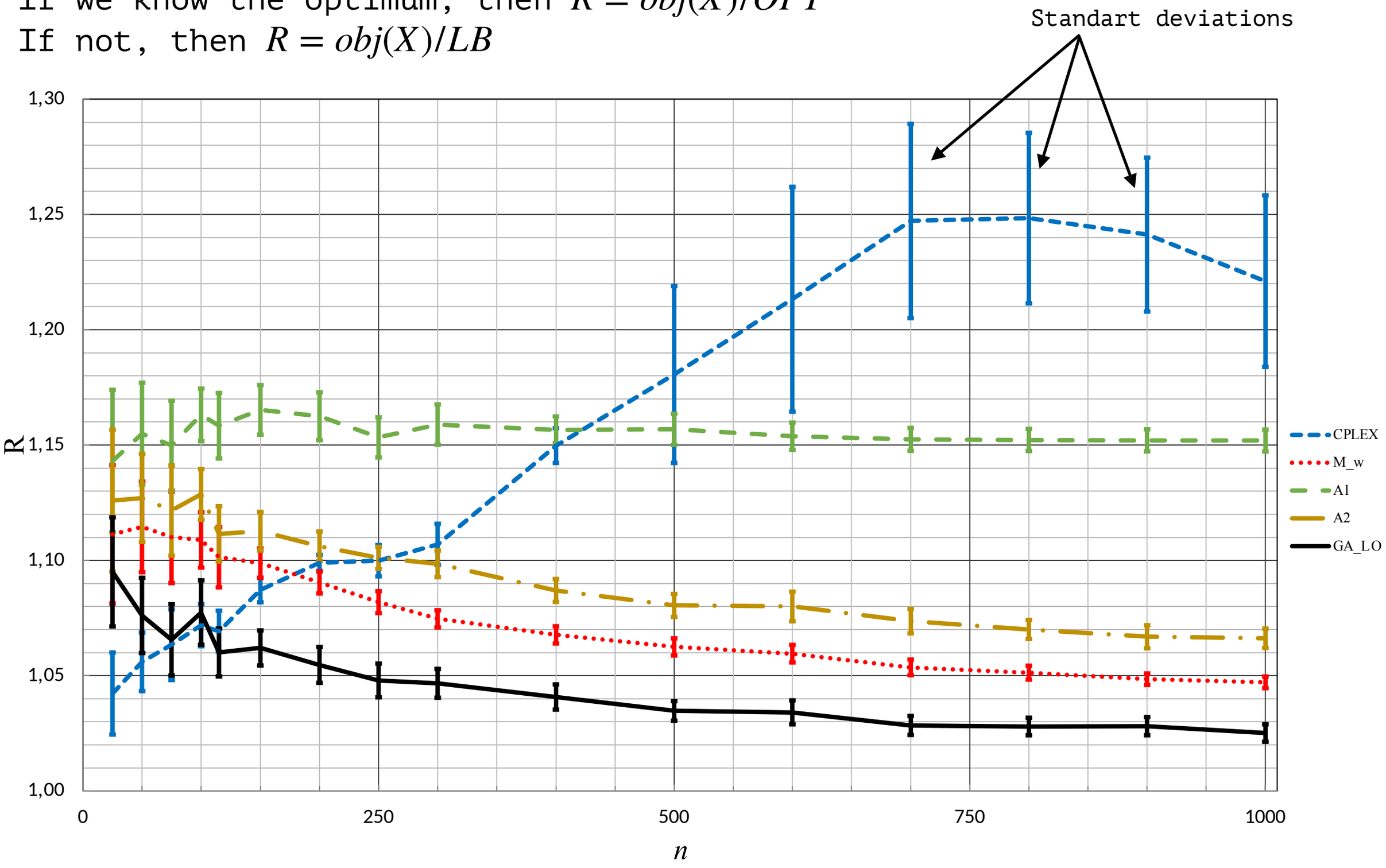
If we know the optimum, then R is the ratio $R = obj(X)/OPT$, where OPT is the minimum packing length and $obj(X)$ is the length of the packing built by the algorithm $X \in \{A1, A2, M_w, GA\_LO\}$.

On the other hand, if CPLEX fails to find the optimum during the allotted time, then we set $R = obj(X)/LB$, where $obj(X)$ is the objective's value of the approximate solution built by the algorithm $X \in \{CPLEX, A1, A2, M_w, GA\_LO\}$, and LB is the lower bound of the optimum yielded by CPLEX.

Randomly generated data

Results of the numerical experiment
with the arbitrary 2-BCs

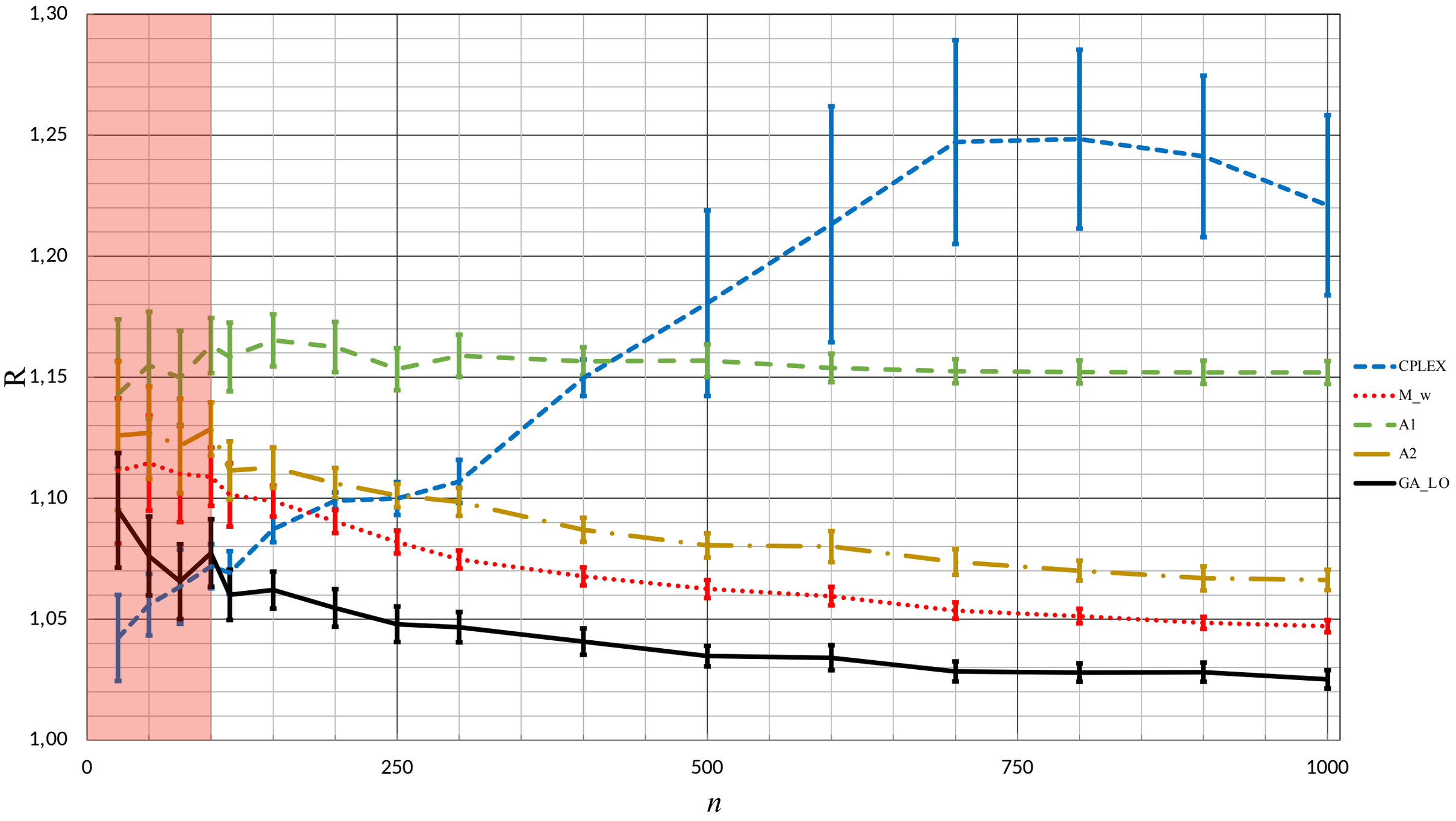
If we know the optimum, then $R = obj(X)/OPT$
If not, then $R = obj(X)/LB$



Randomly generated data

Results of the numerical experiment
with the arbitrary 2-BCs

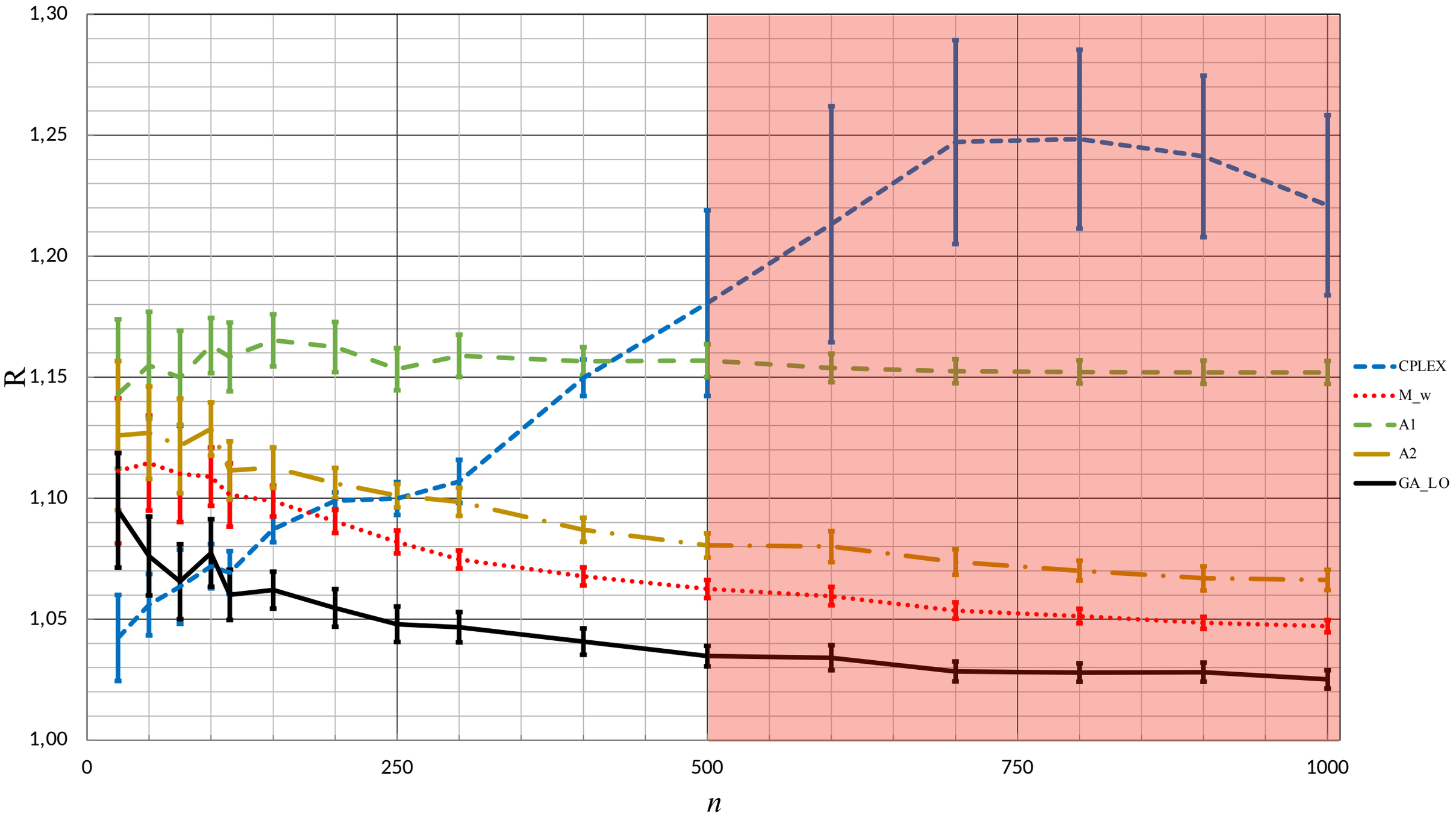
If we know the optimum, then $R = obj(X)/OPT$
If not, then $R = obj(X)/LB$



Randomly generated data

Results of the numerical experiment
with the arbitrary 2-BCs

If we know the optimum, then $R = obj(X)/OPT$
If not, then $R = obj(X)/LB$



Randomly generated data

Results of the numerical experiment
with the arbitrary 2-BCs

n	CPLEX			M_W			M1_W			A1			A2			GA_LO		
	min	max	av	min	max	av	min	max	av	min	max	av	min	max	av	min	max	av
25	0	3	**1,0**	0	5	2,8	1	6	3,9	0	6	3,6	1	7	3,2	0	5	2,4
50	0	5	**2,8**	1	9	5,8	3	10	6,5	2	11	7,9	2	10	6,4	0	7	3,9
75	0	9	**4,8**	1	14	8,3	1	14	8,9	5	16	11,4	1	15	9,2	0	12	5,0
100	1	12	**7,2**	2	15	10,9	2	16	11,3	10	23	16,4	9	19	12,9	2	15	7,8
115	0	11	7,9	2	16	11,6	2	18	12,0	11	24	18,2	5	19	12,8	0	14	**6,9**
150	10	18	13,1	12	20	14,8	12	20	15,1	18	31	24,7	12	25	16,9	5	16	**9,3**
200	16	25	19,8	14	23	18,1	14	23	18,2	24	42	32,5	16	27	21,3	6	19	**11,0**
250	18	39	25,0	16	26	20,5	16	27	20,8	27	47	38,4	19	34	25,3	5	20	**12,1**
300	23	51	32,2	18	27	22,5	18	27	22,5	37	61	47,8	23	38	29,7	8	24	**14,1**
400	48	87	60,0	21	33	27,1	21	33	27,3	52	71	62,7	28	50	34,9	8	29	**16,4**
500	61	188	89,9	24	39	31,1	24	39	31,3	60	95	78,1	30	54	40,1	9	31	**17,3**
600	70	255	128,1	28	50	35,7	29	50	36,0	76	108	92,3	32	72	48,1	11	38	**20,5**
700	78	258	173,1	27	49	37,5	28	49	37,7	93	120	106,8	37	66	51,6	12	28	**19,9**
800	92	287	198,6	34	56	41,0	34	56	41,1	102	136	121,7	37	73	56,0	11	39	**22,4**
900	105	335	217,4	34	55	43,7	34	56	43,9	119	159	137,0	41	88	60,4	14	54	**25,4**
1000	112	343	221,0	39	64	47,1	39	64	47,4	129	175	152,1	53	93	66,3	13	45	**25,2**

Absolute errors of the algorithms

Randomly generated data

Results of the numerical experiment with the arbitrary 2-BCs

n	CPLEX	M_W	M1_W	A1	A2	GA_LO
25	206	0,55	0,23	0,07	0,27	**0,00**
50	288	0,31	0,26	0,23	0,46	**0,00**
75	292	0,44	0,33	0,58	0,72	**0,00**
100	300	0,67	0,50	1,28	0,91	**0,00**
115	300	0,62	0,50	1,20	0,84	**0,01**
150	300	1,00	0,79	1,64	1,39	**0,01**
200	300	1,62	1,33	2,85	2,38	**0,01**
250	300	2,69	2,19	4,48	3,70	**0,02**
300	300	3,72	3,21	6,44	5,25	**0,03**
400	300	7,20	6,17	11,91	9,63	**0,06**
500	301	11,75	10,10	18,14	14,73	**0,09**
600	301	18,06	15,43	26,75	21,20	**0,13**
700	301	25,63	22,47	37,25	29,10	**0,18**
800	301	38,12	34,17	49,83	38,45	**0,24**
900	301	49,92	44,07	63,99	49,01	**0,30**
1000	303	68,60	61,77	80,43	61,00	**0,37**

The running time of the algorithms

Randomly generated data

Results of the numerical experiment with the **big** 2-BCs

n	CPLEX		M_W		M1_W		A1		A2		GA_LO	
	Rav	Rsd	Rav	Rsd	Rav	Rsd	Rav	Rsd	Rav	Rsd	Rav	Rsd
25	**1,003**	0,011	1,006	0,017	1,008	0,021	1,009	0,019	1,007	0,015	1,013	0,019
50	**1,003**	0,008	1,006	0,01	1,007	0,01	1,005	0,009	1,006	0,01	1,009	0,01
75	**1,004**	0,007	1,005	0,008	1,006	0,008	1,006	0,008	1,006	0,008	1,007	0,008
100	**1,003**	0,004	1,004	0,005	1,005	0,005	1,004	0,005	1,005	0,006	1,006	0,005
250	1,207	0,014	**1,202**	0,015	1,202	0,015	1,202	0,015	1,202	0,015	1,203	0,015
500	1,242	0,025	1,203	0,009	1,203	0,009	**1,203**	0,009	1,203	0,01	1,203	0,01
750	1,237	0,01	1,201	0,009	1,201	0,009	**1,201**	0,009	1,201	0,009	1,201	0,009
1000	1,234	0,01	1,202	0,007	1,202	0,007	1,202	0,007	1,202	0,007	**1,202**	0,007

All 2-BCs are big

R_{av} is the mean value

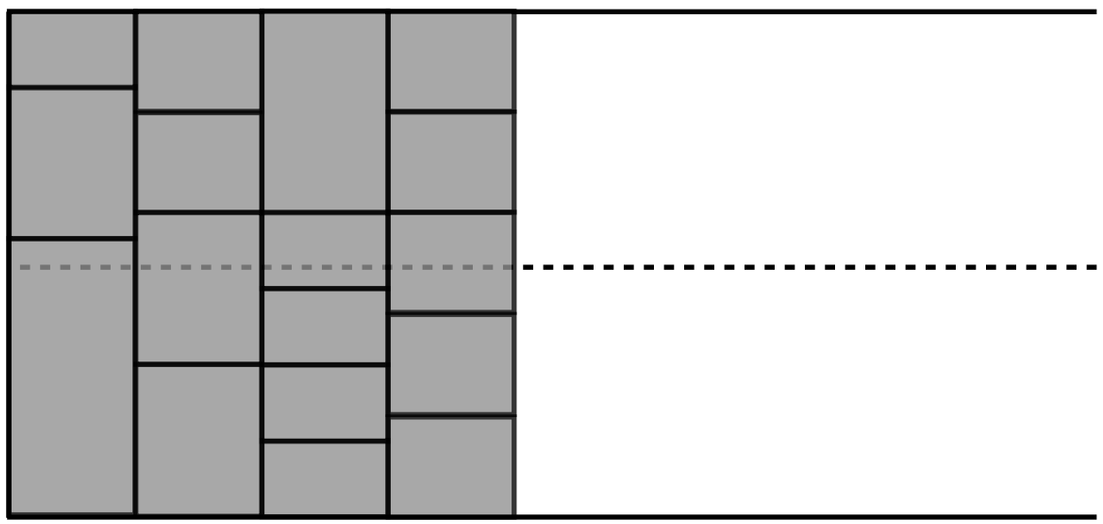
R_{sd} is the standard deviation

n	CPLEX		M_W		M1_w		A1		A2		GA_LO	
	Rav	Rsd	Rav	Rsd	Rav	Rsd	Rav	Rsd	Rav	Rsd	Rav	Rsd
25	**1,005**	0,013	1,013	0,019	1,058	0,039	1,008	0,016	1,008	0,016	1,022	0,017
50	**1,017**	0,027	1,022	0,027	1,061	0,031	1,02	0,027	1,02	0,027	1,025	0,025
75	**1,018**	0,02	1,023	0,021	1,07	0,029	1,025	0,022	1,025	0,022	1,024	0,02
100	**1,022**	0,043	1,025	0,044	1,068	0,052	1,026	0,042	1,026	0,043	1,026	0,042
250	1,235	0,018	**1,216**	0,014	1,269	0,01	1,218	0,015	1,218	0,015	1,216	0,014
500	1,26	0,021	1,212	0,01	1,263	0,008	1,216	0,01	1,216	0,01	**1,21**	0,009
750	1,247	0,011	1,207	0,009	1,259	0,006	1,211	0,009	1,211	0,01	**1,206**	0,009
1000	1,247	0,008	1,205	0,008	1,258	0,005	1,21	0,008	1,21	0,008	**1,204**	0,009

All 2-BCs are big and non-increasing

BPP benchmarks

In **[7]** authors randomly generated 3840 instances. We use these instances because they are available online as well as their optimal solutions.



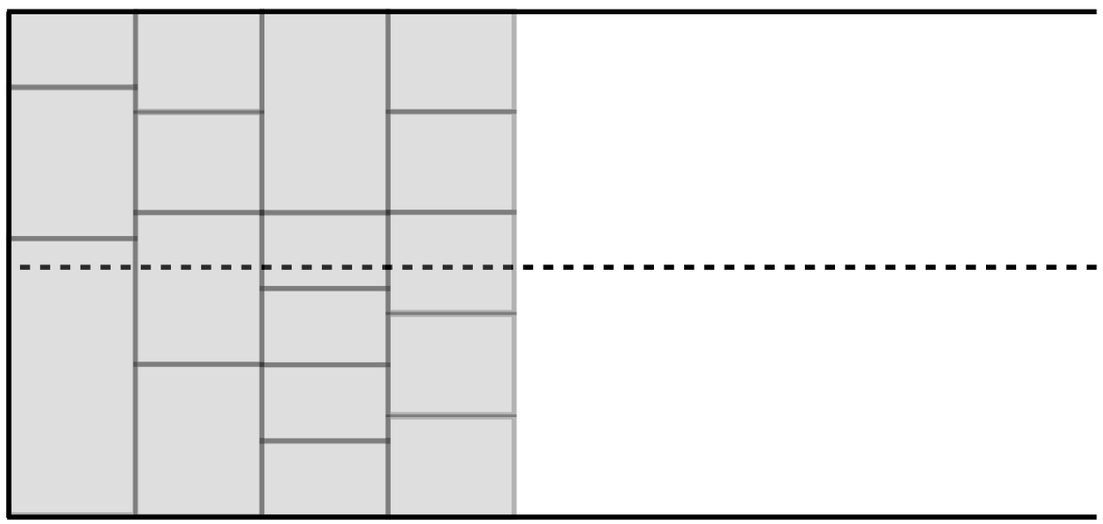
Let N be the number of the bins in **optimal packing**

n_1 n_2 n_3 n_4

[7] **Delorme M., Iori M., Martello S.: Bin Packing and Cutting Stock Problems: Mathematical Models and Exact Algorithms. European Journal of Operational Research 225(1), 1-20 (2016)**

BPP benchmarks

In **[7]** authors randomly generated 3840 instances. We use these instances because they are available online as well as their optimal solutions.



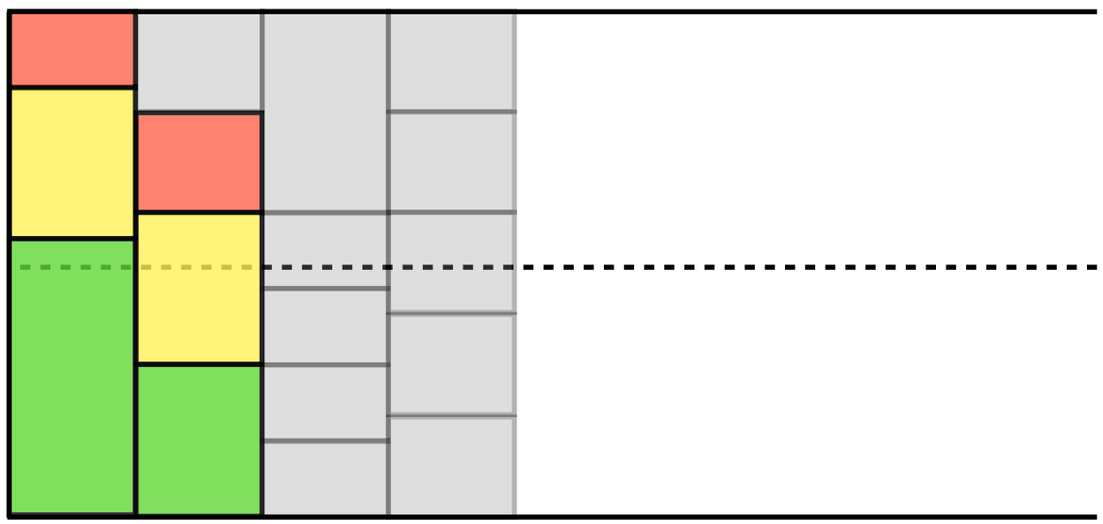
Let N be the number of the bins in **optimal packing**

n_1 n_2 n_3 n_4

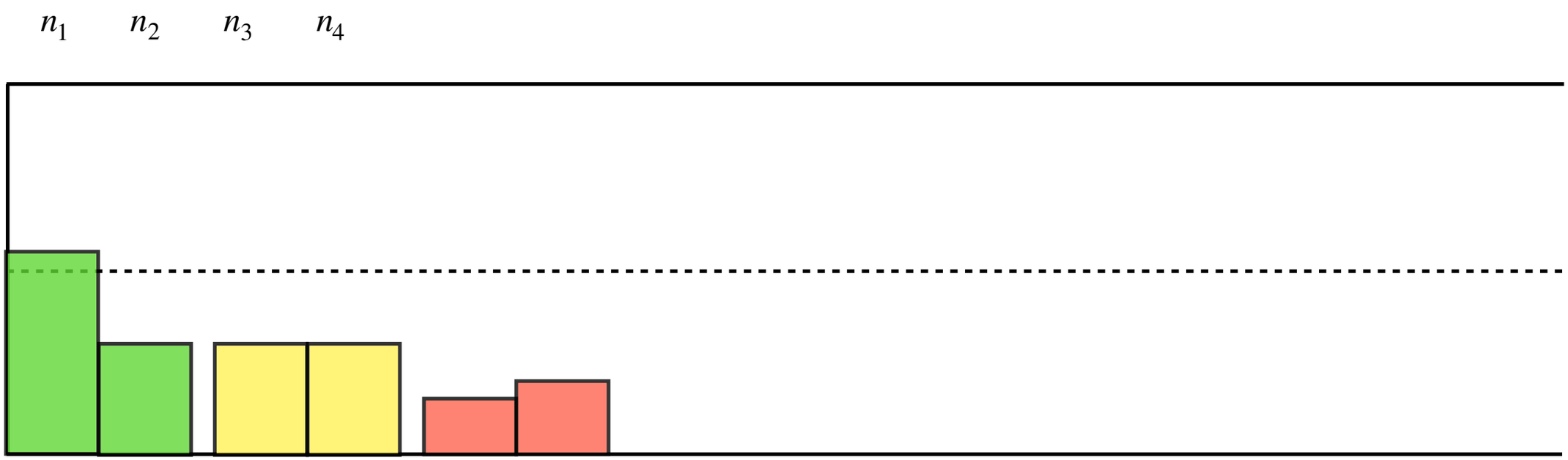
[7] **Delorme M., Iori M., Martello S.: Bin Packing and Cutting Stock Problems: Mathematical Models and Exact Algorithms. European Journal of Operational Research 225(1), 1-20 (2016)**

BPP benchmarks

In [7] authors randomly generated 3840 instances. We use these instances because they are available online as well as their optimal solutions.



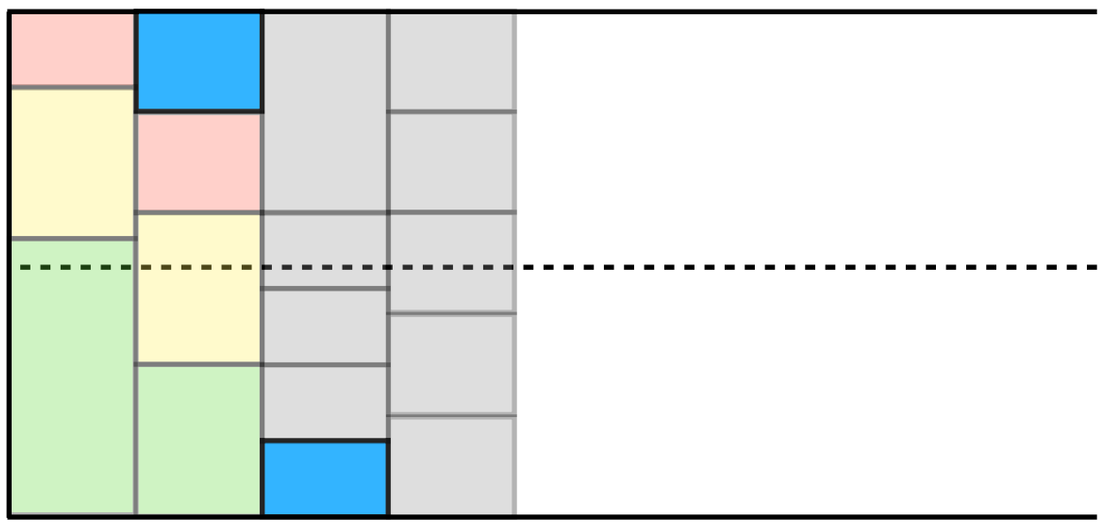
n_1 items in the first and the second bins create n_1 2-BCs



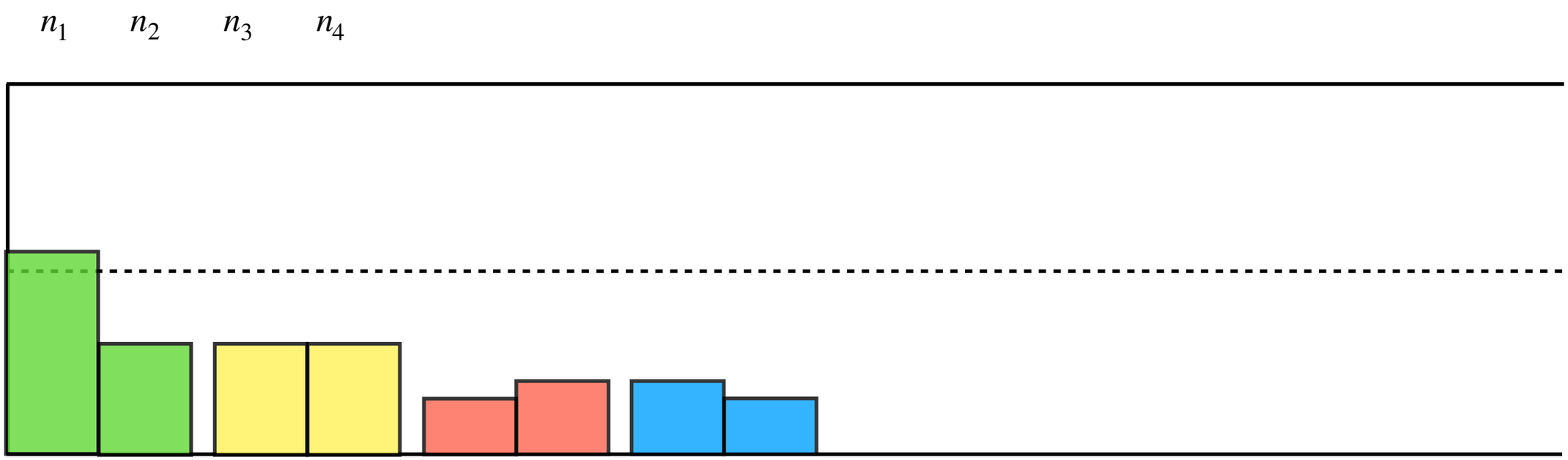
[7] Delorme M., Iori M., Martello S.: Bin Packing and Cutting Stock Problems: Mathematical Models and Exact Algorithms. European Journal of Operational Research 225(1), 1-20 (2016)

BPP benchmarks

In [7] authors randomly generated 3840 instances. We use these instances because they are available online as well as their optimal solutions.



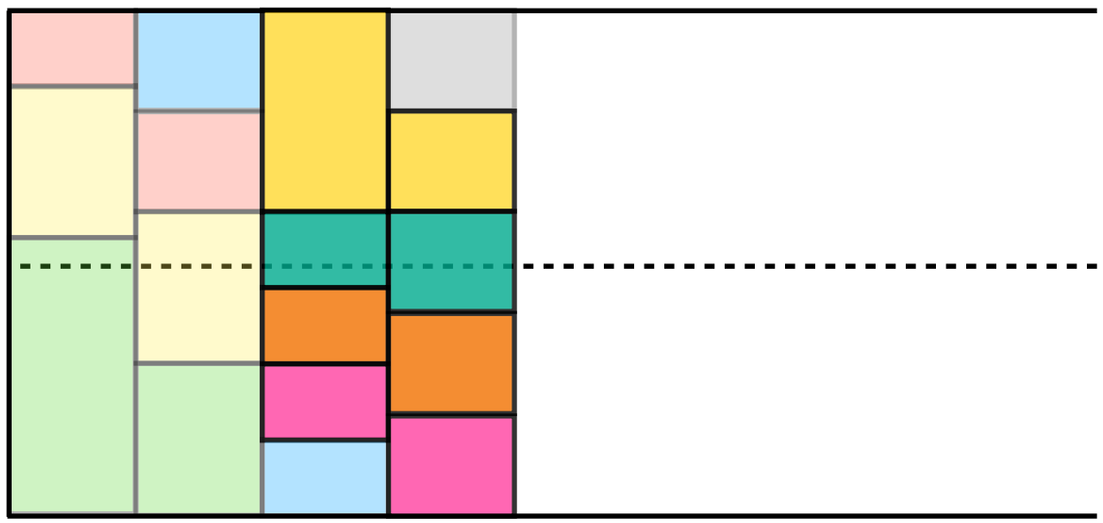
$n_2 - n_1$ items from the second and the third bins form the next 2-BCs



[7] Delorme M., Iori M., Martello S.: Bin Packing and Cutting Stock Problems: Mathematical Models and Exact Algorithms. European Journal of Operational Research 225(1), 1-20 (2016)

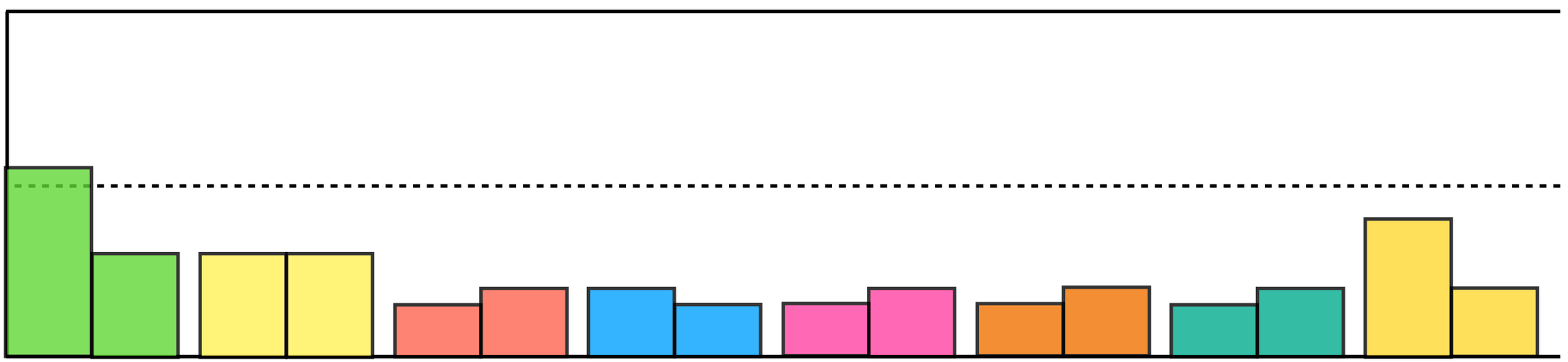
BPP benchmarks

In [7] authors randomly generated 3840 instances. We use these instances because they are available online as well as their optimal solutions.



The remaining $n_3 - (n_2 - n_1)$ items of the third bin create 2-BCs with items from the fourth bin

n_1 n_2 n_3 n_4

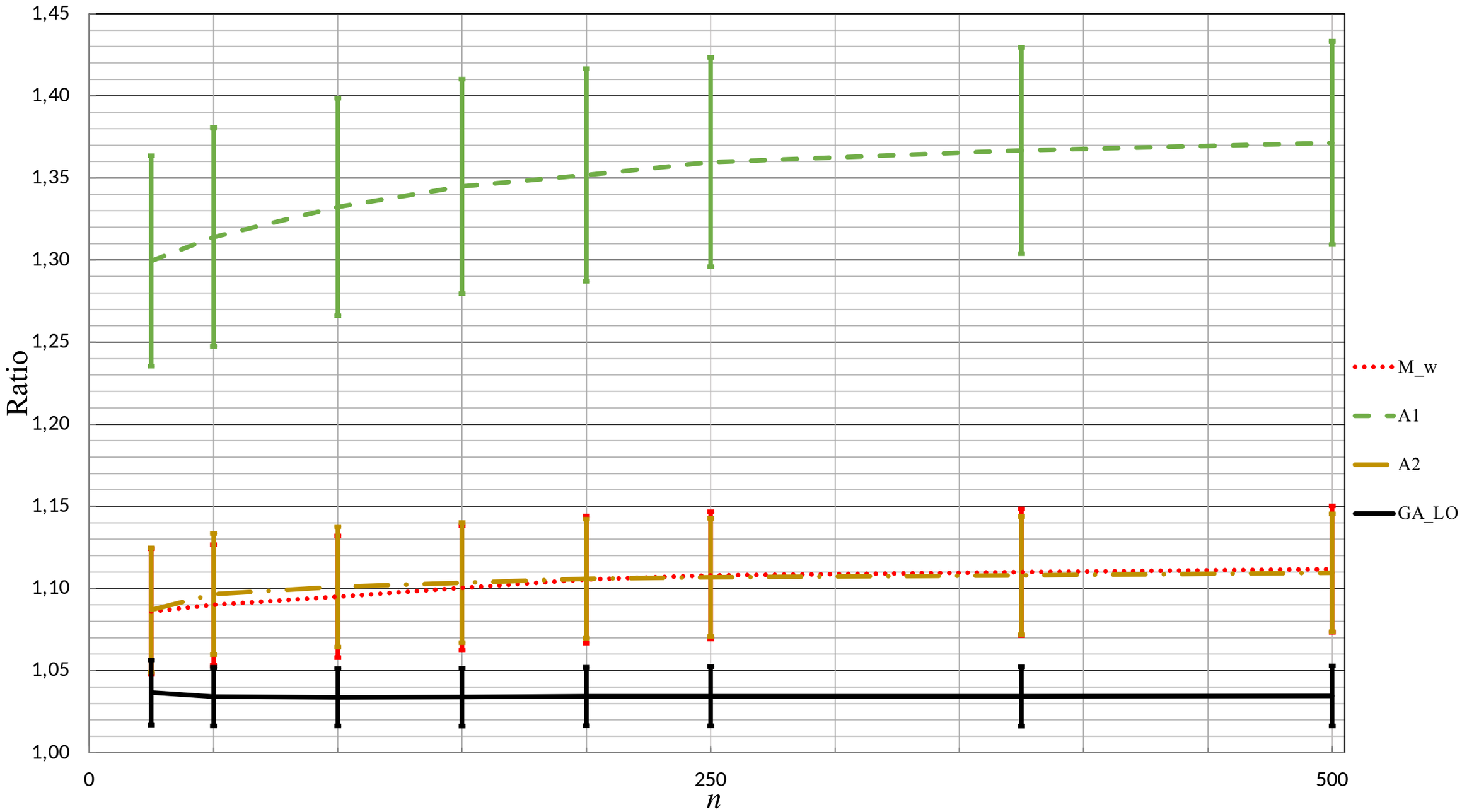


[7] Delorme M., Iori M., Martello S.: Bin Packing and Cutting Stock Problems: Mathematical Models and Exact Algorithms. European Journal of Operational Research 225(1), 1-20 (2016)

BPP benchmarks

Results of the numerical experiment

For each $n \in \{25,50,100,150,200,250,375,500\}$, we generate 480 different test instances with known optimal solutions. And now let ratio be $obj(X)/OPT$, where OPT is the optimum and $obj(X)$ is the objective's value found by the algorithm $X \in \{A1,A2,M_w,GA\_LO\}$.



BPP benchmarks

Results of the numerical experiment

n	M_W		M1_W		A1		A2		GA_LO	
	max	av	max	av	max	av	max	av	max	av
25	7	1,9	11	3,7	12	7,0	6	1,9	5	**0,8**
50	11	4,1	24	9,0	24	15,0	12	4,7	6	**1,4**
100	26	9,2	49	16,2	48	33,7	23	9,6	14	**2,9**
150	39	15,2	74	24,4	72	52,0	33	14,5	23	**4,4**
200	54	22,1	99	34,1	98	68,6	43	20,1	27	**6,2**
250	62	26,4	124	38,4	118	90,3	63	24,1	39	**7,5**
375	91	39,6	183	58,2	177	138,2	79	37,3	55	**10,7**
500	120	54,7	249	80,1	235	181,8	103	52,8	83	**15,6**

Upper bounds on the absolute errors

n	M_W		M1_W		A1		A2		GA_LO	
	#	%	#	%	#	%	#	%	#	%
25	132	27,5	124	25,8	6	1,3	136	28,3	**196**	**40,8**
50	74	15,4	70	14,6	3	0,6	69	14,4	**126**	**26,3**
100	68	14,2	67	14,0	0	0,0	65	13,5	**71**	**14,8**
150	**71**	**14,8**	71	14,8	0	0,0	66	13,8	57	11,9
200	**59**	**12,3**	58	12,1	0	0,0	53	11,0	50	10,4
250	**62**	**12,9**	62	12,9	0	0,0	59	12,3	48	10,0
375	**67**	**14,0**	67	14,0	0	0,0	66	13,8	46	9,6
500	**60**	**12,5**	59	12,3	0	0,0	54	11,3	51	10,6

The number and percentage of optimal solutions

Conclusion

The numerical experiment shows the high efficiency of the greedy algorithm with the preliminary lexicographic ordering of the 2-BCs (GA_LO), which significantly outperforms other algorithms in accuracy and runtime.

In future research, we are planning to test various other heuristics and metaheuristics, for which a priori accuracy estimates are not necessarily known.

Thank you for your attention!