Heuristics with Local Improvements for Two-processor Scheduling Problem with Energy Constraint and Parallelization

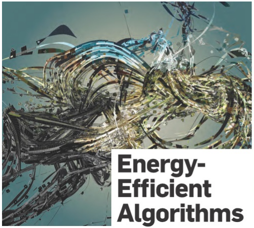
M. Sakhno, Yu. Zakharova

June 19, 2023

Omsk Department of Sobolev Institute of Mathematics SB RAS
The research is supported by RSF grant 22-71-10015.

Motivation (Parallel and Multiprocessor Jobs)

- Parallel jobs require more than one processor at the same time.
- Some jobs can not be performed asynchronously on modern computers. Such situation takes place in multiprocessor graphics cards, where the memory capacity of one processor is not sufficient.
- Many computer systems offer some kinds of parallelism. The energy efficient scheduling of parallel jobs arises in testing and reliable computing, parallel applications on graphics cards, computer control systems and others.



Report Structure

- Problem Statement
- Previous Research
- Greedy heuristics and lower bounds
- Local improvements and experimental evaluation
- Conclusion and Further Research

Speed Scaling Scheduling: $P2|size_j, energy|\sum C_j$

Processors and Jobs

$m = 2$ speed-scalable processors

$\mathcal{J} = \{1, \ldots, n\}$ is the set of jobs:
V_j is the processing volume (work) of job j
$size_j$ is the number of processors required by job j
$W_j := \frac{V_j}{size_j}$ is the work on one processor

E is the energy budget

Parameters

Preemption and migration are characterized for the systems with single image of the memory.
Non-preemptive instances arise in systems with distributed memory.

Homogeneous Model in Speed-scaling

If a processor runs at speed s then the energy consumption is s^α units of energy per time unit, where $\alpha > 1$ is a constant (practical studies show that $\alpha \leq 3$).

It is supposed that a continuous spectrum of processor speeds is available.

The aim is to find a feasible schedule with the minimum total completion time so that the energy consumption is not greater than a given energy budget.

Previous Research: Classic

Makespan

Drozdowski (2009): poly for parallel jobs, pmtn, r_j
approx for parallel jobs, r_j
Brucker (2000), Du, Leung (1989): parallel jobs: NP-hard, strongly NP-hard for prec

Total Completion Time

Lee and Cai (1999): parallel jobs: strongly NP-hard
Schwiegelshohn et. al. (1998), J. Turek et. al. (1994): approximation algorithms for parallel jobs
Hoogeveen (1994): single-mode jobs: NP-hard
Cai (1998): 2-approximation algorithm for single-mode jobs

Previous Research: Energy

Makespan

Pruhs, van Stee (2007), Bunde (2009): poly for single processor, r_j
approx for multiple processors, r_j
Bampis et.al. (2014): approx for prec, r_j

Total Completion Time

Pruhs et. al. (2008), Bunde (2009): poly for single processor
Shabtay, Kaspi (2006): approx for multiple processors

Parallel jobs

Kononov, Zakharova (2017-2022): NP-hardness and approx
Kong F. et. al. (2011): level-packing algorithms
Li K. (2012): partitioning-scheduling-supplying

Convex Program (KKT-conditions)

Two-processor Jobs

$$\frac{1}{2}\sum_{i=1}^{n}(n-i+1)p_{\pi_i} \to \min,$$

$$\sum_{i=1}^{n}(V_{\pi_i})^{\alpha} p_{\pi_i}^{1-\alpha} = E.$$

Single-processor Jobs

$$\sum_{j\in\mathcal{J}} C_j(\pi) = \sum_{j=1}^{\frac{n}{2}}\left(\frac{n}{2}-j+1\right)\left(p_{\pi_{2j-1}}+p_{\pi_{2j}}\right) \to \min,$$

$$\sum_{j\in\mathcal{J}} {p_j}^{1-\alpha}(V_j)^{\alpha} = E.$$

NP-hardness

Even-Odd Partition Problem

$A = \{a_1, a_2, \ldots, a_{2n_0}\}$ is the ordered set such that
$\sum_{a_i \in A} a_i = 2C$, $a_i < a_{i+1},\ i = 1, \ldots, 2n_0 - 1$
$a_{2i+1} > 3a_{2i}$ for $i = 1, \ldots, n_0 - 1$.
Question: whether A can be partitioned into two subsets A_1 and A_2

$$\sum_{a_i \in A_1} a_i = \sum_{a_i \in A_2} a_i = C,\ |A_1| = |A_2| = n_0,$$

A_1 contains only one element from each pair $a_{2i-1},\ a_{2i},\ i = 1, \ldots, n_0$.

Theorem

Problem $P2|size_j, energy|\sum C_j$ is NP-hard.

Greedy Heuristic (Algorithm 1)

Scheme

Step 1: Given an instance I of $P2|size_j, energy|\sum C_j$, we generate the instance I' with fully-parallelizable jobs, construct optimal schedule S' for jobs, corresponding to non-decreasing order of volumes V_j, and find optimal durations p_j.
Step 2: Calculate processing times of jobs for instance I: $\frac{2p_j}{size_j}$, $j = 1, \ldots, n$. Assign job j to the first available processor if j requires one processor or to the two processors when both of them are available if j is a two-processor job while keeping the order of jobs in non-decreasing of volumes V_j.

Lemma

$\sum C_j(S') \leq \sum C_j^*$.

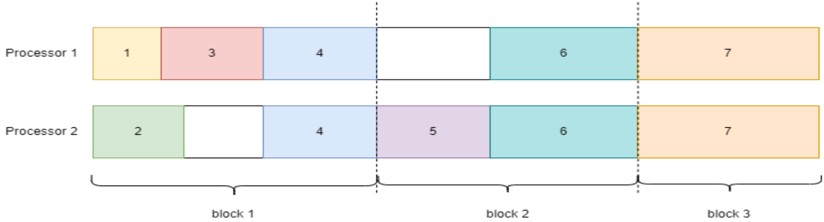
Theorem

A 2-approximate schedule can be found by Algorithm 1 in $O(n \log n)$ time for scheduling problem $P2|size_j, energy|\sum C_j$.

Local improvements between blocks

1. Find blocks.

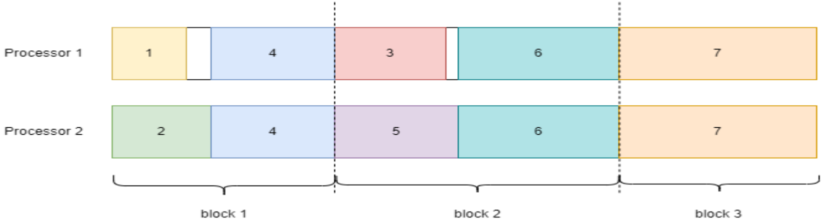
2. If a block consists of an odd number of single-processor jobs, move the last job to the next block if possible.



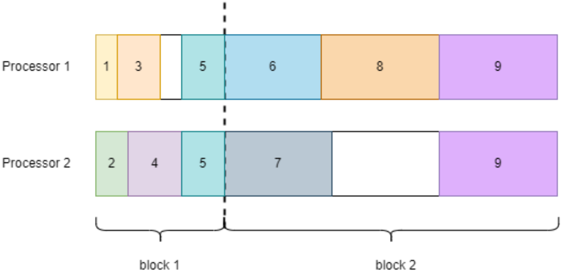
Local improvements between blocks

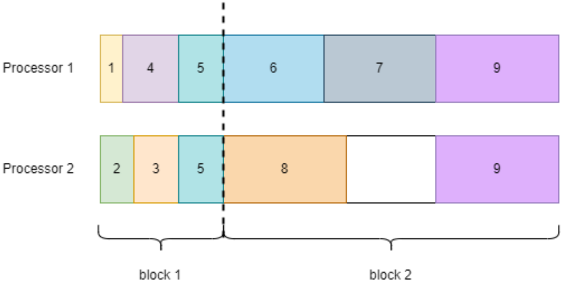
1. Find blocks.

2. If a block consists of an odd number of single-processor jobs, move the last job to the next block if possible.



Local improvements inside blocks





Greedy Heuristic with Local Improvements (Algorithm 2)

Step 1. Construct a schedule by Greedy Heuristic and find blocks in the solution.

Step 2. Consequently apply the local improvements between blocks.

Step 3. Apply local improvements inside blocks to the given solution.

Step 4. Return the found solution.

Test instances

- alpha (1.5, 2.0, 2.5, 3.0)
- jobs count (50, 100)
- small jobs probability (0.0, 0.3, 0.5, 0.7, 1.0)
- single jobs probability (0.3, 0.5, 0.7)
- series (11, 12, 21, 22)

Instances count in series = 30

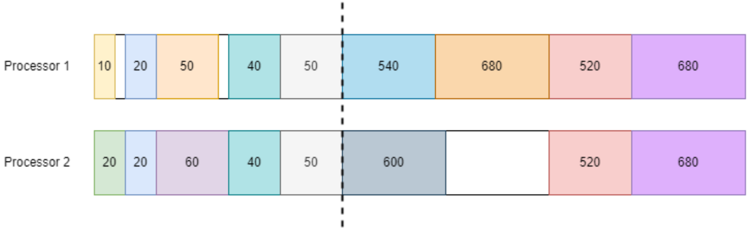
Series 12

$$\mathbf{SMALL}_1 = (10, 20, 30, 40, 50, 60, 70, 80, 90, 100)$$

$$\mathrm{SMALL}_2 = (10, 11, 12, 13, 14, 15, 16, 17, 18, 19)$$

$$\mathrm{LARGE}_1 = (200, 275, 350, 425, 500, 575, 650, 725, 800, 875)$$

$$\mathbf{LARGE}_2 = (520, 540, 560, 580, 600, 620, 640, 660, 680, 700)$$



Test results for series 12

$\alpha = 1.5$, $n = 50$

single	**small**	E_{GH}	D_{GH}	E_{GH-LI}	D_{GH-LI}
0.3	0.0	1.14	0.14	**1.03**	0.24
0.3	0.3	1.74	0.42	**1.32**	0.51
0.3	0.5	3.16	1.06	**2.03**	0.71
0.3	0.7	4.53	2.20	**2.59**	1.70
0.3	1.0	7.26	5.49	**2.28**	0.96
0.5	0.0	1.41	0.10	**1.29**	0.16
0.5	0.3	2.79	0.47	**2.12**	0.34
0.5	0.5	3.90	0.86	**2.51**	0.52
0.5	0.7	6.46	1.71	**3.74**	1.25
0.5	1.0	9.49	6.51	**3.49**	1.18
0.7	0.0	1.96	0.08	**1.83**	0.12
0.7	0.3	3.20	0.20	**2.62**	0.18
0.7	0.5	4.47	0.71	**3.32**	0.57
0.7	0.7	6.49	2.47	**4.20**	1.74
0.7	1.0	8.93	3.18	**3.68**	0.71
avg		4.46	1.71	2.54	0.73

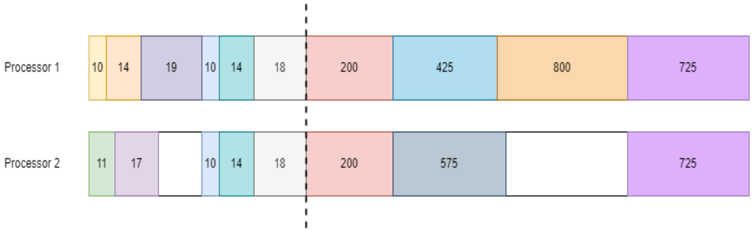
Series 21

$$\text{SMALL}_1 = (10, 20, 30, 40, 50, 60, 70, 80, 90, 100)$$

$$\mathbf{SMALL}_2 = (10, 11, 12, 13, 14, 15, 16, 17, 18, 19)$$

$$\mathbf{LARGE}_1 = (200, 275, 350, 425, 500, 575, 650, 725, 800, 875)$$

$$\text{LARGE}_2 = (520, 540, 560, 580, 600, 620, 640, 660, 680, 700)$$



Test results for series 21

$\alpha = 1.5$, $n = 50$

single	**small**	E_{GH}	D_{GH}	E_{GH-LI}	D_{GH-LI}
0.3	0.0	4.17	1.22	**2.16**	0.78
0.3	0.3	5.86	3.44	**3.30**	2.26
0.3	0.5	8.21	6.68	**4.01**	2.88
0.3	0.7	9.35	14.94	**5.71**	10.22
0.3	1.0	1.07	0.10	**0.89**	0.14
0.5	0.0	5.57	2.04	**2.43**	0.86
0.5	0.3	7.89	3.92	**3.75**	1.60
0.5	0.5	8.65	8.42	**4.39**	1.81
0.5	0.7	10.28	8.05	**6.49**	5.81
0.5	1.0	1.58	0.13	**1.46**	0.21
0.7	0.0	5.91	1.05	**3.45**	0.65
0.7	0.3	7.16	3.13	**4.41**	1.21
0.7	0.5	8.09	5.42	**5.42**	2.90
0.7	0.7	12.85	13.69	**8.06**	13.48
0.7	1.0	2.08	0.05	**1.94**	0.08
avg		6.58	4.82	3.86	2.99

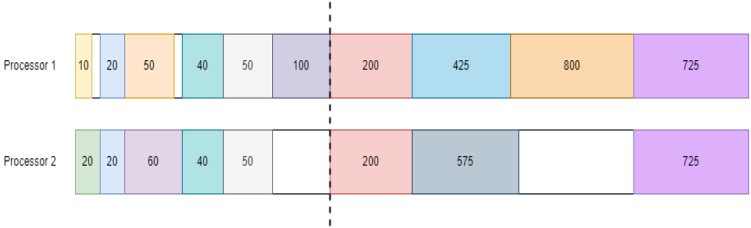
Series 11

$$\mathbf{SMALL}_1 = (10, 20, 30, 40, 50, 60, 70, 80, 90, 100)$$

$$\mathrm{SMALL}_2 = (10, 11, 12, 13, 14, 15, 16, 17, 18, 19)$$

$$\mathbf{LARGE}_1 = (200, 275, 350, 425, 500, 575, 650, 725, 800, 875)$$

$$\mathrm{LARGE}_2 = (520, 540, 560, 580, 600, 620, 640, 660, 680, 700)$$



Test results for series 11

$\alpha = 1.5$, $n = 50$

single	small	E_{GH}	D_{GH}	E_{GH-LI}	D_{GH-LI}
0.3	0.0	4.17	1.22	**2.16**	0.78
0.3	0.3	6.16	3.10	**3.47**	2.24
0.3	0.5	8.78	6.09	**3.83**	2.26
0.3	0.7	9.48	10.39	**5.05**	5.35
0.3	1.0	7.26	5.49	**2.28**	0.96
0.5	0.0	5.57	2.04	**2.43**	0.86
0.5	0.3	8.18	4.16	**3.77**	1.41
0.5	0.5	9.87	6.80	**4.23**	1.61
0.5	0.7	11.41	5.63	**5.69**	3.44
0.5	1.0	9.49	6.51	**3.49**	1.18
0.7	0.0	5.91	1.05	**3.45**	0.65
0.7	0.3	7.58	3.50	**4.35**	1.04
0.7	0.5	8.75	4.24	**5.29**	2.21
0.7	0.7	12.45	8.67	**6.76**	5.76
0.7	1.0	8.93	3.18	**3.68**	0.71
avg		8.27	4.80	4.00	2.03

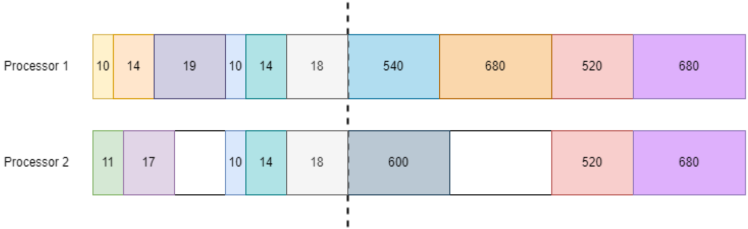
Series 22

$$\text{SMALL}_1 = (10, 20, 30, 40, 50, 60, 70, 80, 90, 100)$$

$$\mathbf{SMALL}_2 = (10, 11, 12, 13, 14, 15, 16, 17, 18, 19)$$

$$\text{LARGE}_1 = (200, 275, 350, 425, 500, 575, 650, 725, 800, 875)$$

$$\mathbf{LARGE}_2 = (520, 540, 560, 580, 600, 620, 640, 660, 680, 700)$$

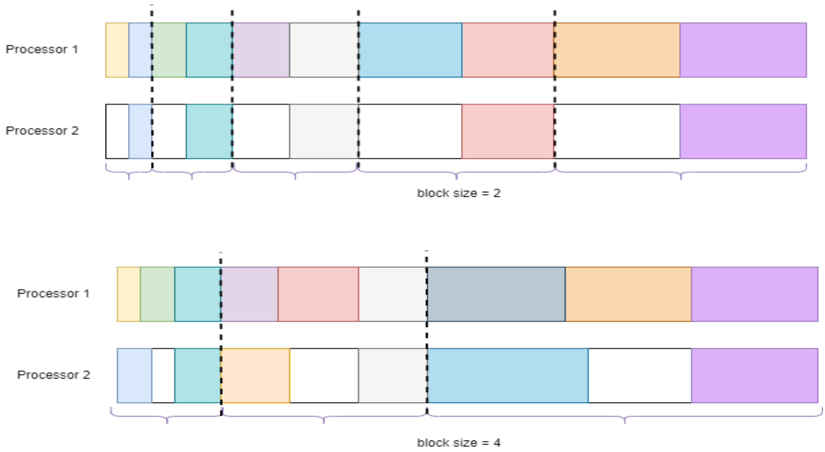


Test results for series 22

$\alpha = 1.5$, $n = 50$

single	**small**	E_{GH}	D_{GH}	E_{GH-LI}	D_{GH-LI}
0.3	0.0	1.14	0.14	**1.03**	0.24
0.3	0.3	1.48	0.36	**1.23**	0.59
0.3	0.5	2.46	0.64	**2.01**	0.92
0.3	0.7	3.37	2.51	**2.65**	2.64
0.3	1.0	1.07	0.10	**0.89**	0.14
0.5	0.0	1.41	0.10	**1.29**	0.16
0.5	0.3	2.35	0.32	**2.06**	0.44
0.5	0.5	3.01	0.55	**2.50**	0.79
0.5	0.7	5.02	1.72	**4.09**	2.39
0.5	1.0	1.58	0.13	**1.46**	0.21
0.7	0.0	1.96	0.08	**1.83**	0.12
0.7	0.3	2.90	0.16	**2.60**	0.22
0.7	0.5	3.90	0.54	**3.36**	0.73
0.7	0.7	5.75	2.61	**4.63**	3.44
0.7	1.0	2.08	0.05	**1.94**	0.08
avg		2.63	0.67	2.24	0.87

Blocks


Processor 1
Processor 2
block size = 2
Processor 1
Processor 2
block size = 4

Test results for blocks

$\alpha = 1.5$, $n = 50$

block size	**small**	E_{GH}	D_{GH}	E_{GH-LI}	D_{GH-LI}
2	0.0	50.71	0.00	**6.40**	0.24
4	0.0	25.63	0.00	**5.11**	0.08
6	0.0	17.36	0.00	**4.99**	0.05
8	0.0	13.25	0.00	**4.84**	0.07
10	0.0	10.94	0.00	**5.26**	0.05
2	0.3	50.92	0.00	**9.13**	1.01
4	0.3	25.88	0.00	**7.47**	0.57
6	0.3	17.65	0.00	**7.17**	0.27
8	0.3	13.56	0.00	**6.66**	0.24
10	0.3	11.34	0.00	**7.09**	0.23
2	0.5	51.14	0.01	**11.53**	1.41
4	0.5	26.13	0.01	**9.15**	0.81
6	0.5	17.93	0.01	**8.40**	0.94
8	0.5	13.86	0.01	**7.85**	0.44
10	0.5	11.77	0.03	**8.72**	0.30
2	0.7	51.46	0.01	**15.14**	2.38
4	0.7	26.51	0.02	**11.33**	1.26
6	0.7	18.35	0.02	**10.28**	1.30
8	0.7	14.33	0.02	**9.71**	0.53
10	0.7	12.01	0.02	**8.44**	0.20
2	1.0	50.77	0.00	**7.38**	0.45
4	1.0	25.71	0.00	**5.96**	0.19
6	1.0	17.45	0.00	**5.75**	0.13
8	1.0	13.38	0.00	**5.70**	0.08
10	1.0	11.07	0.00	**6.14**	0.07
avg		23.96	0.00	7.82	0.53

Conclusion

Test Conclusions

parameter	best result
alpha	1.5
jobs count	100
small jobs probability	0.3
single jobs probability	0.3
blocks	10
series	22

The average relative deviation from the lower bound is not greater than 8% over all considered series.
The difference between the record values of GH and GH_LI is statistically significant at level less than 0.05 on all series.

Comparison with single-processor case

The comparison with the single-processor case showed that even partial parallelization can lead to improvement.

Further Research

- Comparison with commercial solvers.
- Investigation of lower bounds.
- More accurate selections in local improvements.
- Developing meta heuristics.

Thank you for your attention!

General Convex Model

$$\sum_{j \in J} x_{jik} \leq 1, i \in I, k \in K,$$

$$\sum_{k \in K} w_{jk} = 1, j \in J,$$

$$\sum_{i \in I} x_{jik} = size_j w_{jk}, k \in K, j \in J,$$

$$T_{jk}^{f} \geq T_{jk}^{st}, k \in K, j \in J,$$

$$T_{jk}^{f} - T_{jk}^{st} = w_{jk} p_j, k \in K, j \in J,$$

$$\sum_{k \in K} (T_{jk}^{f} - T_{jk}^{st}) \geq p_j, j \in J,$$

$$T_{jk}^{st} \geq T_{j'k'}^{f} - T_{max}(2 - x_{jik} - x_{j'ik'}),$$

$$j \neq j' \in J, i \in I, k' < k \in K, k \neq 1.$$

$$s_j p_j \geq W_j, j \in J.$$

$$\sum_{j \in J} size_j W_j (s_j)^{\alpha - 1} \leq E.$$

Comparison with Gurobi Solver

Instance	Algorithm			Gurobi Presolve		Gurobi Record (12 threads)			%	LB, %
	LB	**Obj**	**%**	**Obj**	**Time, sec**	**LB**	**Obj**	**Time, sec**		
0.5-0.5-11	1173.5	1421.3	21.1	1553.15	14	1333.07	1333.07*	1270 (13292)	6.6	11.9
0.5-0.5-12	694.1	786.1	13.2	987.74	21	763.65	763.65*	18357 (34095)	2.9	9.1
0.5-0.5-21	812.4	1011.6	24.5	1337.68	21	981.68	981.68*	17810 (58895)	3.0	17.2
0.5-0.5-22	1984.5	2330.1	17.4	2544.72	38	2231.22	2231.22*	1795 (6000)	4.4	11.0
-1-0.5-block	1682.3	2470.9	46.8	2442.63	24	1970.97	1970.97*	1060 (5555)	25.3	14.6
0.3-0.0-11	7556.34	7768.93	2.8	10042.25	48	7699.46	7699.46*	645(32978.68)	0.90	1.85
0.3-0.0-12	7904.26	8521.14	7.8	10247.32	47	8384.59	8384.59*	630 (51781.12)	1.62	5.72
0.3-0.0-21	4184.31	4650.28	11.1	4733.77	101	3910.78	4407.69	12191	5.50	5.06
0.3-0.0-22	6594.20	7250.91	9.9	7669.93	83	5059.43	7077.13	120	2.45	6.82
-1-0.0-block	7434.83	9141.18	22.9	10254.9	81	5857.43	8010.14	4851	14.12	7.18
0.7-0.7-11	145.85	161.99	11.0	200.05	62	82.01	158.27	5080	2.35	7.84
0.7-0.7-12	241.54	309.48	28.1	691.85	10	239.54	301.85	445	2.52	19.97
0.7-0.7-21	58.18	97.99	68.4	99.83	8	96.43	96.43*	22095 (24395)	1.61	39.65
0.7-0.7-22	409.79	480.05	17.1	547.65	14	439.97	465.47	18880	3.13	11.96
-1-0.7-block	1253.93	1809.16	44.2	2697.62	25	1436.99	1436.99*	250 (7955)	25.89	12.73

Comparison with single-processor case

$\alpha = 1.5,\ n = 50$

single	**small**	E_{A2}	E_1	MAX_w	MAX_b	AVG_w	AVG_b	$COUNT_b$
0.3	0.0	2.16	4.76	–	3.94	–	2.55	30
0.3	0.3	3.47	6.15	0.43	4.77	0.43	2.70	29
0.3	0.5	3.83	7.75	–	6.75	–	3.69	30
0.3	0.7	5.05	9.66	1.15	7.93	1.15	4.43	29
0.3	1.0	2.28	5.22	–	4.02	–	2.80	30
0.5	0.0	2.43	5.07	–	4.04	–	2.64	30
0.5	0.3	3.77	6.61	–	4.21	–	2.77	30
0.5	0.5	4.23	8.16	0.20	6.29	0.20	3.87	29
0.5	0.7	5.69	10.29	–	7.00	–	4.24	30
0.5	1.0	3.49	5.57	–	3.86	–	1.97	30
0.7	0.0	3.45	5.15	–	3.47	–	1.80	30
0.7	0.3	4.35	6.60	–	4.06	–	2.28	30
0.7	0.5	5.29	8.24	–	5.80	–	2.87	30
0.7	0.7	6.76	10.51	3.45	7.31	2.59	3.93	28
0.7	1.0	3.68	5.67	–	3.91	–	1.88	30