Didelės skaičiavimo apimties skaitinių algoritmų analizė ir optimizavimas

Rima Kriauzienė (VU MII)

Doktorantūros metai – 2015 m. – 2019 m.
4-ieji doktorantūros metai
Ataskaita už 3-iuosius doktoratūros metus

Vadovas – prof. habil. dr. R. Čiegis (VGTU),
Konsultantas – prof. dr. J. Žilinskas (VU MII).

Mokslo kryptis: Informatika (09 P).

Spalio 24 d., 2018 m., Vilnius, Lietuva

Doktorantūros planas 2017-10-01 – 2018-10-01

Tyrimo tikslas:

- Didelės skaičiavimo apimties skaitinių algoritmų analizė ir optimizavimas.

Tyrimo objektas:

- Didelės skaičiavimo apimties skaitiniai lygiagretieji algoritmai.

Tyrimo uždaviniai

- Specialių kraštinių sąlygų koeficientų radimo uždavinių sprendimo analizė ir išvadų parengimas.
- Lygiagrečiųjų algoritmų tokiems uždaviniams spręsti sudarymas, išplečiamumo analizė.

Doktorantūros planas 2017-10-01 – 2018-10-01

Dalyvavimas konferencijose, kitose doktorantų mobilumo veiklose

- DAMSS: 9th international workshop on data analysis methods for software systems, November 30–December 2, 2017, Druskininkai.
- MMA2018: 23nd international conference, May 29-June 1, 2018, Sigulda, Latvia.

Žiemos mokykla

3rd NESUS Winter School and PhD Symposium 2018, 22nd-25th January 2018, Zagreb, Croatia.
Pranešimas – Numerical analysis and optimization of parallel algorithms for problems with big computational costs.

http://nesusws.irb.hr/images/BookofAbstracts.pdf

Mokslinių tyrimų ir disertacijos rengimo etapai

Čiegis, R.s; Starikovičius, V.; Margenov, S.; **Kriauzienė, R.**. A comparison of accuracy and efficiency of parallel solvers for fractional power diffusion problems // Parallel Processing and Applied Mathematics: 12th international conference, PPAM 2017, Lublin, Poland, September 1013, 2017. *Lecture Notes in Computer Science*, 2018, vol. 10777, p. 79-89. DOI: 10.1007/978-3-319-78024-5_8.

Recenzavimo būsenoje

- Čiegis, R.; Starikovičius, V.; Margenov, S.; **Kriauzienė, R**. A scalability analysis of different parallel solvers for 3D fractional power diffusion problems. *Concurrency and computation: practice and experience*.
- **Kriauzienė, R.**; Bugajev, A.; Čiegis, R. A three-level parallelisation scheme and application to the optimisation problems. *IEEE Transactions on Parallel and Distributed Systems*.

Doktorantūros planas 2018-10-01 – 2019-10-01

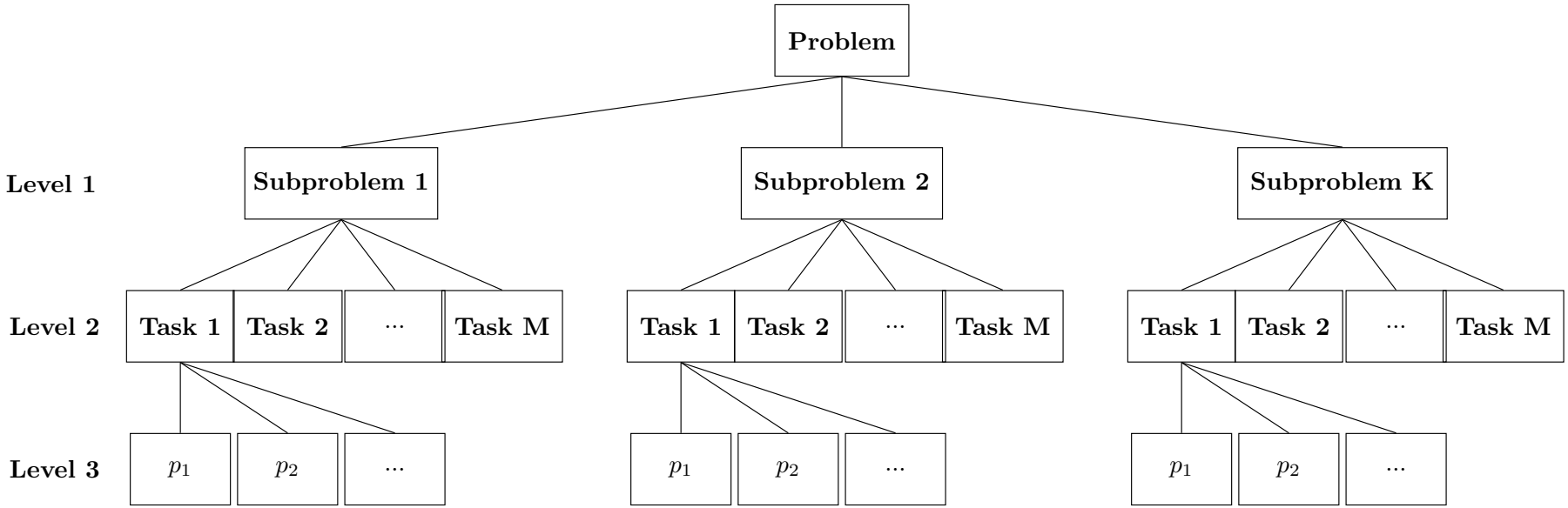
Dalyvauti konferencijose:

- DAMSS: 10th international workshop on data analysis methods for software systems, November 29–December 1, 2018, Druskininkai.

Mokslinio tyrimo vykdymas:

- Daktaro disertacijos parengimas ir svarstymas padalinyje – 2019 m. 06 mėn.
- Daktaro disertacijos gynimas – 2019 m. 09 mėn.

The problem and parallelization strategies

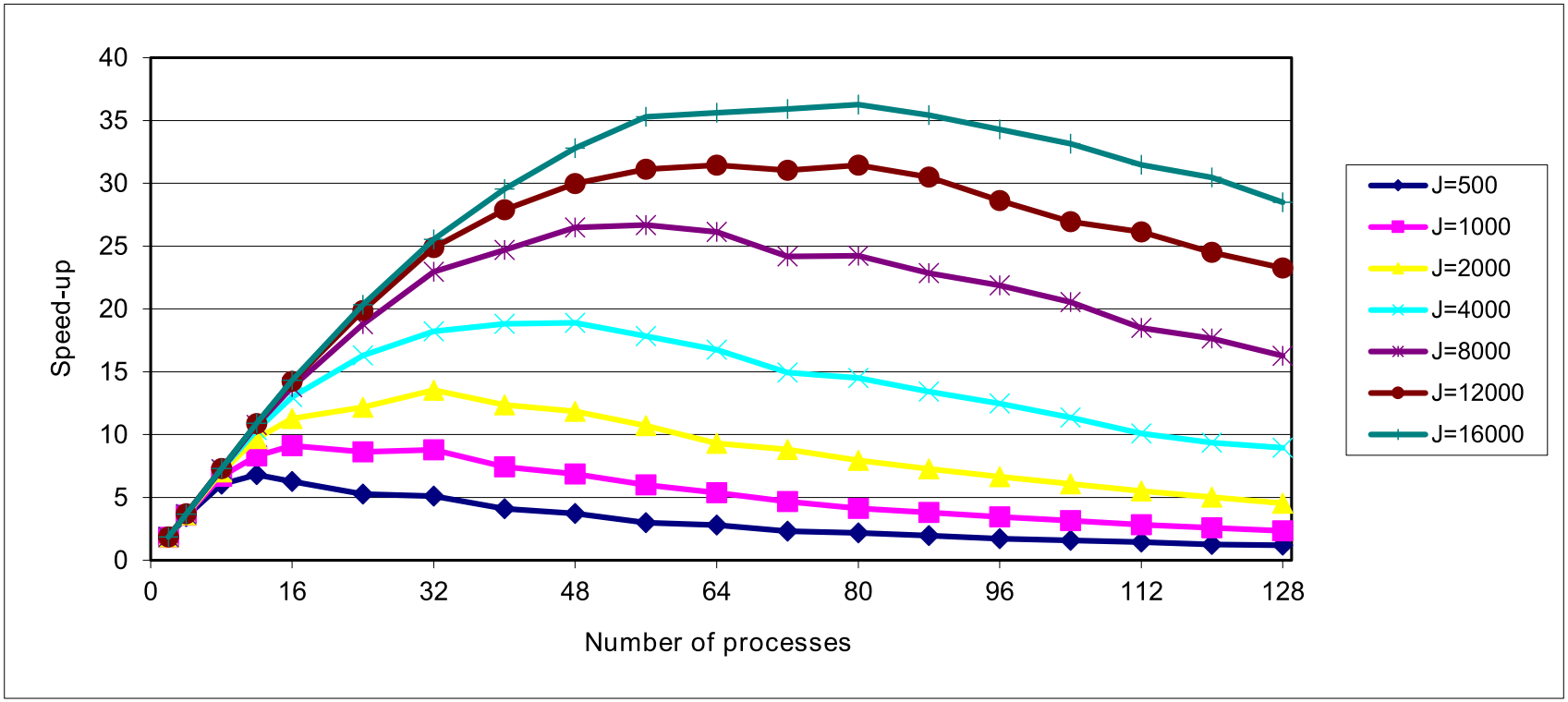


As an example, it can be M different PDEs $L_j u = 0, 1 \leq j \leq M$, which are approximated numerically with $L_j^h U = 0, 1 \leq j \leq M$ with solutions u^h. Solutions depend on parameters $q_1, q_2, \ldots, q_m$.

The optimization problem

$$\min_{q_1, q_2, \ldots, q_m} F(u_1^h(q_1, q_2, \ldots, q_m), \ldots, u_M^h(q_1, q_2, \ldots, q_m))$$

Workload distribution



We propose hybrid multi–level approach, which improves the scalability.

Workload distribution

- We consider the following main minimisation problem: find the optimal value k_0 of task blocks

$$T_0(k_0) = \min_{1 \le k \le K} T_B\left(P/k\right)/\Gamma(k), \tag{1}$$

- $\Gamma(k) = k\gamma_k$, $0 < \gamma_k \le 1$, where $T_B(p)$ defines the optimal time for solving one block of M tasks using p processes:

$$T_B(P) = \min_{(p_1,\ldots,p_M)\in S} \max_{1 \le m \le M} t_m(p_m), \tag{2}$$

where a set S of feasible processors distributions is defined as

$$S = \{(p_1, \ldots, p_M): \quad p_m \le P_m,\ m = 1, \ldots, M, \quad p_1 + \ldots + p_M \le P\}.$$

ALGORITHM 1: Distribution of P processes between M tasks

Set $p[m] = 1$, for $m = 1, \ldots, M$
$P = P - M$
Compute $t_m(p[m])$, for $m = 1, \ldots, M$
stop $= 0$
while $P > 0$ & $stop == 0$ **do**
    find j such that $t_j(p[j]) = \max\limits_{1 \le m \le M} t_m(p[m])$
    if $p[j] == P_j$ **then**
        stop $= 1$
    else
        $p[j] = p[j] + 1$; $P = P - 1$;
    end
end

- The number of processes cannot exceed the number after which the speed-up begin to drop.
- The number of processes is limited by efficiency requirement, which means it is not allowed increase the number of processes per task v_j if it makes the calculations efficiency smaller than the selected constant E_{min}.

$$\frac{t_j(1)}{p\, t_j(p)} \ge E_{min}.$$

Linear Schrödinger equation

$$
\begin{aligned}
& i\frac{\partial u}{\partial t} + \frac{\partial^2 u}{\partial x^2} = 0, \ x \in (a,b), t \in [0,T] \\
& u(x,0) = u_0(x) \\
& L_l u(a) = 0, L_r u(b) = 0.
\end{aligned} \tag{3}
$$

The boundary conditions approximation [1]

$$
\partial_n u = -e^{-i\frac{\pi}{4}} \left(\left(\sum_{k=1}^{\frac{m+1}{2}} q_k \right) u - \sum_{k=1}^{\frac{m-1}{2}} q_{k+1} q_{k+(m+3)/2} \varphi_k \right), \tag{4}
$$

where $\partial_n u$ is the normal derivative, the number of parameters $m \in \mathbb{N}$ is odd and φ_k is obtained from

$$
\frac{d\varphi_k(x,t)}{dt} + q_{k+(m+3)/2}\varphi_k(x,t) = u(x,t), \ x = a, b, \ k = 1, \ldots, [m/2].
$$

[1] A. Bugajev, R. Čiegis, R. Kriauzienė, T. Leonavičienė and J. Žilinskas (2017). On the Accuracy of Some Absorbing Boundary Conditions for the Schrödinger Equation. Mathematical Modelling and Analysis, 22(3):408-423.

Optimization problem

Then we formulate an optimization problem

$$\min_{q_1,q_2,\ldots,q_m} \max_{1\leq j\leq M} \|u_j - u_j^h(q_1, q_2, \ldots, q_m)\|_\infty$$

Notes

- Assuming that u_j is known, each value of functional that is being minimized requires to solve M different equations.
- All M equations can be solved independently.

First level of parallel algorithm

As a local optimizer Nelder-Mead algorithm is used.

During each iteration we can have these different scenarios

- Reflection (one point: f_R)
- Expansion (two points: f_R, f_e)
- Contraction (two points: f_R, f_c)

Note: in this case $K = 3$, $\gamma_1 = 1$, $\gamma_2 = 0.75$, $\gamma_3 = 2/3$.

Second level of parallel algorithm

We compute functional

$$\max_{1\le j\le M}\|u_j - u_j^h(q_1, q_2, \ldots, q_m)\|_\infty = F(q_1, q_2, \ldots, q_m), \tag{5}$$

where u^h approximation error should be small comparing to F.

Tasks

U_j with different j can be computed independently, computation of maximum is small comparing to computations of finding solutions and computational errors. This leads to efficient parallel calculation of solutions.

Third level of parallel algorithm

The system of linear equations

$$\begin{cases} b_0 x_0 + c_0 x_1 = d_0, \\ a_i x_{i-1} + b_i x_i + c_i x_{i+1} = d_i, \quad i = 1, \ldots, N-2 \\ a_{N-1} x_{N-2} + b_{N-1} x_{N-1} = d_{N-1}, \end{cases} \tag{6}$$

where a_i, b_i, c_i, d_i are complex numbers

- Linear equations with tridiagonal matrix are solved using Wang's algorithm.

Improvement on the second level

Third level let us to perform the workload balancing at the second level of our algorithm. We assign different numbers of processes for different problems with different computational costs.

Benchmarks

Table: Benchmarks with different sizes

Benchmark 1		Benchmark 2		Benchmark 3	
Eq.	$J \times N$	Eq.	$J \times N$	Eq.	$J \times N$
1	$\mathbf{8000} \times \mathbf{40000}$	1	$8000 \times \mathbf{20000}$	1	8000×10000
2	4000×20000	2	4000×20000	2	$\mathbf{2000} \times 20000$
3	2000×20000	3	$\mathbf{4000} \times \mathbf{10000}$	3	2000×10000
4	2000×10000	4	2000×10000	4	$\mathbf{1000} \times 20000$

The results of the first benchmark

p	16	32	64	96	128	128
		$k = 1$		$k = 2$	$k = 3$	$K = 1$
Eq. 1	10	22	**50**	34	29	**56**
Eq. 2	3	5	8	8	7	8
Eq. 3	2	3	4	4	4	4
Eq. 4	1	2	2	2	2	2
Model time	11.145	5.784	3.614	2.742	2.272	3.605
Exp. time	11.003	5.394	**3.608**	2.719	2.308	**3.600**
Speed-up	12.68	25.86	38.66	51.31	60.44	38.75

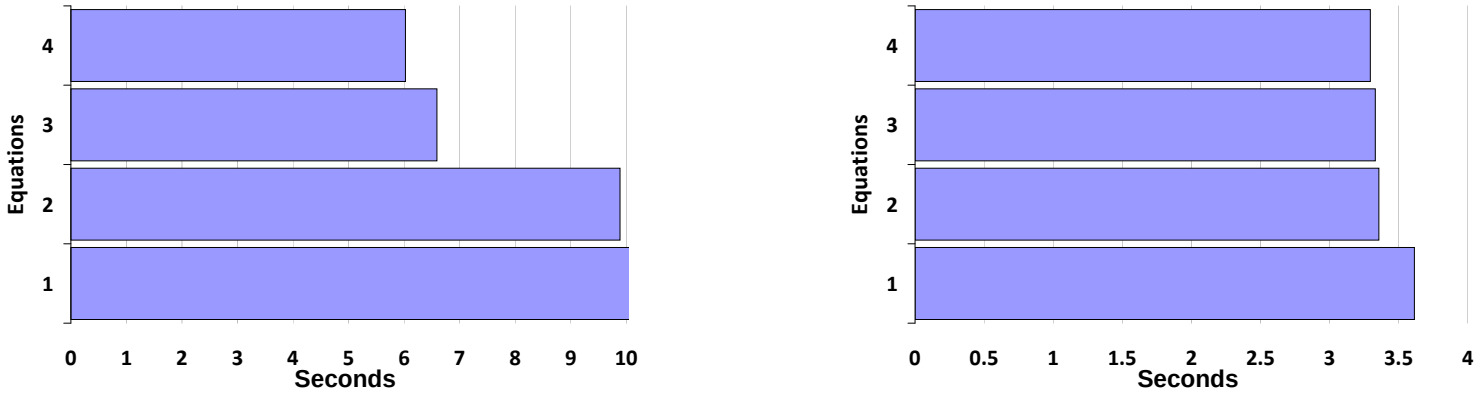


Figure: Benchmark 1 T values with $p = 16$(left) and $p = 64$(right)

The results of benchmark 1 with $E_{min} > 0$

p	128		
E_{min}	0.75 $k = 3$	0.8 $k = 3$	0.6 $K = 1$
Eq. 1	26	19	42
Eq. 2	7	5	8
Eq. 3	4	3	4
Eq. 4	2	1	2
Model time	2.45	3.17	3.84
Exp. time	2.49	3.08	3.76
Speed-up	56.03	45.37	37.11

4 problems

$$u_1(t,x) = \frac{\exp\left(-i\pi/4\right)}{\sqrt{4t-i}} \exp\left(\frac{ix^2 - 6x - 36t}{4t-i}\right). \tag{7}$$

$x \in [-5,5]$, $t \in [0,0.8]$. $J \times N = 8000 \times 4000$.

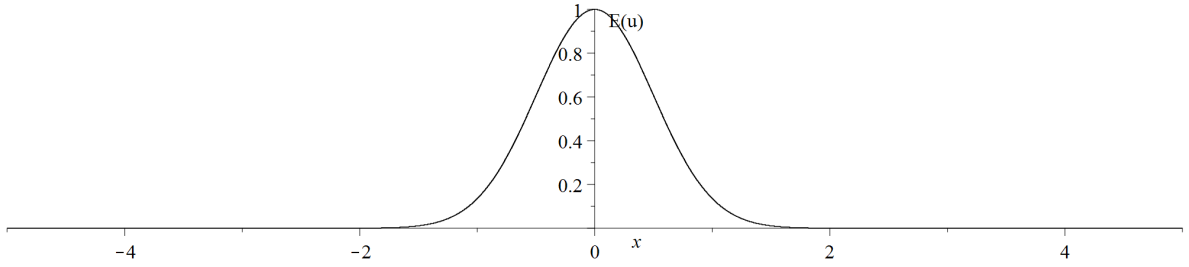
$$u_2(t,x) = \frac{1}{\sqrt[+]{1+it/\alpha}} \exp\left(ik(x - x^{(0)} - kt) - \frac{(x - x^{(0)} - 2kt)^2}{4(\alpha + it)}\right), \tag{8}$$

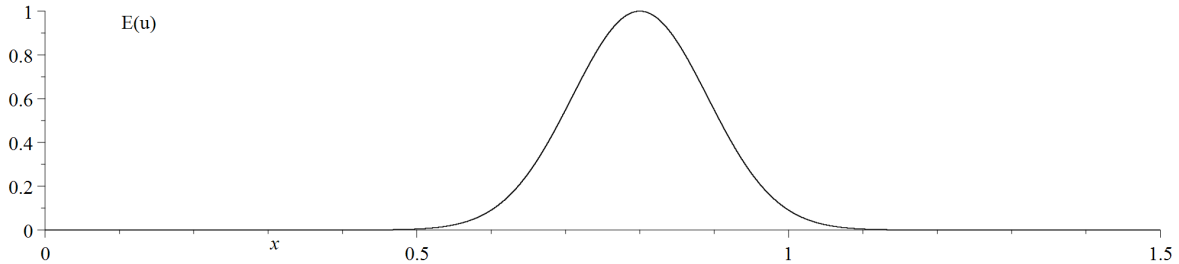
where $k = 100, \alpha = 1/120, x^{(0)} = 0.8$. $x \in [0,1.5]$, $t \in [0,0.04]$, $J \times N = 12000 \times 4000$

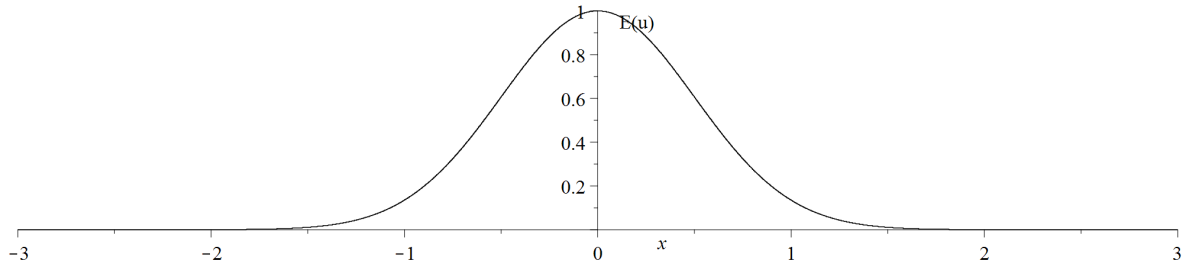
The solution is defined by (7), $x \in [-10,10]$, $t \in [0,2]$. We use uniform discretisation grid $J \times N = 16000 \times 10000$.

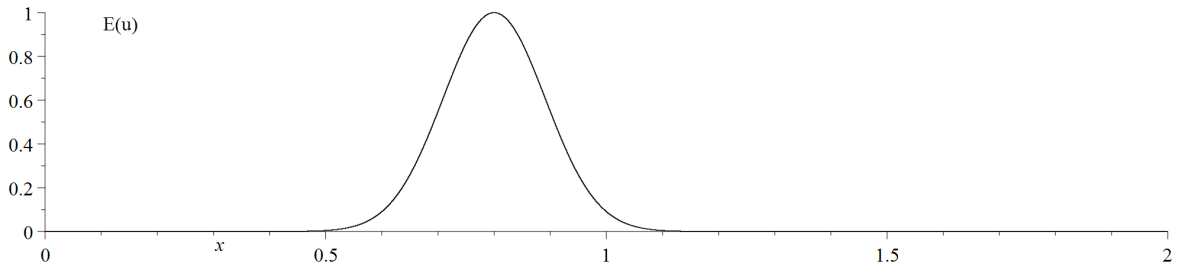
The solution is defined by (8), $k = 100, \alpha = 1/120, x^{(0)} = 0.8$. $x \in [0,2]$, $t \in [0,0.08]$. We use uniform discretisation grid $J \times N = 16000 \times 8000$.

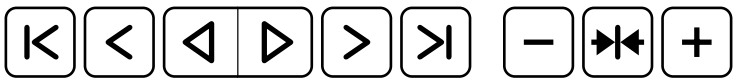
Solutions of example











Conclusions

1. Comparing to one level parallelization the proposed algorithm with three levels greatly expands the number of processes that can be used.

2. Workload balancing was analysed, model-based approach performs balancing well enough for practical purposes. The model prediction times are close to times of the real computational experiments.

3. We propose the heuristic with parameter E_{min} which guarantees that the efficiency of calculations on the third level is not lower than the value of E_{min}. This heuristic is not optimal, however, for considered cases we show that it is sufficient.

Ačiū už dėmesį.