Mathematical Modelling and Analysis
Volume 4, 1999, pages 58–69

A PARALLEL ALGORITHM FOR SOLVING ONE PROBLEM OF NONLINEAR OPTICS

R. ČIEGIS[1,2], A. DEMENT'EV[3] and P. RATĖ[4]

[1] *Institute of Mathematics and Informatics,*
Akademijos 4, 2600 Vilnius, Lithuania,
[2] *Vilnius Gediminas Technical University,*
Saulėtekio St. 11, 2054 Vilnius, Lithuania,
[3] *Institute of Physics,*
Goštauto 12, 2600 Vilnius, Lithuania,
[4] *Vytautas Magnus University,*
Vileikos 8, 3035 Kaunas, Lithuania,
E-mail: [2] rc@fm.vtu.lt

Received October 18, 1999

ABSTRACT

This paper deals with a system of nonlinear differential equations, which describe the interaction of two focused laser beams in nonlinear media. The system of equations is approximated by a splitting finite difference scheme. A parallel version of the finite-difference scheme is proposed and the efficiency of this algorithm is investigated. Calculations are performed using clusters of computers, connected via local computer network. The emphasis is made on solving this problem on heterogeneous clusters. In the paper a static distribution scheme is analyzed. The results of several computational experiments are presented. Data redistribution during an initial phase of computation is investigated and the influence of slow communication among the processes is taken into account during this analysis.

1. INTRODUCTION

We consider the problem of two counteracting focused laser beams in non–linear media. It is described by the system of nonlinear equations in $Q =$

$\{(z,r,t): \quad 0<z<L, \quad 0<r<R, \quad 0<t<T\}$:

$$\frac{\partial e_L}{\partial t}+\frac{\partial e_L}{\partial z}-i\mu_L\frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial e_L}{\partial r}\right)+\frac{\alpha_L}{2}e_L=-\Gamma_L\sigma_S e_S$$
$$+i\eta_L\left(|e_L|^2+2|e_S|^2\right)e_L, \tag{1.1}$$
$$\frac{\partial e_S}{\partial t}+\frac{\partial e_S}{\partial z}-i\mu_S\frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial e_S}{\partial r}\right)+\frac{\alpha_S}{2}e_S=-\Gamma_S\sigma_S^*\,e_L$$
$$+i\eta_S\left(2|e_L|^2+|e_S|^2\right)e_S, \tag{1.2}$$
$$\frac{\partial \sigma_S}{\partial t}+a\sigma_S=-\Gamma_\sigma e_L e_S^*, \tag{1.3}$$

where $e_{L,S}$, σ_S are laser, Stokes and hyper sound waves complex amplitudes respectively, Z, t , r are non-dimensional coordinates, $\mu_{L,S}=\frac{1}{2k_{L,S}}$, $k_{L,S}$ are modules of the wave vectors, Γ_L, Γ_S, Γ_σ are the coefficients of nonlinear counter-action, $\alpha_{L,S}$ are laser and Stokes wave absorption coefficients, $\eta_{L,S}$ are the nonlinear refraction indexes.

Boundary conditions are defined on the edge of the region Q:

$$e_L(0,r,t)=e_L^1(r,t), \quad e_S(L,r,t)=e_S^1(r,t), \tag{1.4}$$
$$e_L(z,R,t)=0, \quad e_S(z,R,t)=0,$$
$$r\frac{\partial}{\partial r}e_L(z,0,t)=0, \quad r\frac{\partial}{\partial r}e_S(z,0,t)=0.$$

The initial conditions are given at t=0:

$$e_L(z,r,0)=e_L^0(z,r), \quad e_S(z,r,0)=e_S^0(z,r), \tag{1.5}$$
$$\sigma_S(z,r,0)=\sigma_S^0(z,r,0).$$

2. FINITE DIFFERENCE SCHEME

We solve the system of equations (1.1) - (1.5) using the finite difference method.

2.1. Definition of the discrete grid

Let the computational space Q be covered by a grid $\overline{\omega}_t\times\overline{\omega}_z\times\overline{\omega}_r$

$$\overline{\omega}_t=\{t_n=n\tau,\ n=0,1,\ldots,K,\ t_K=T\},$$
$$\overline{\omega}_r(r_j)=\left\{r_k=kh_j,\ k=0,1,\ldots,M, r_M=R\left(\left(1-\frac{z_j}{f}\right)^2+\frac{z_j^2}{Z_R^2}\right)\right\},$$
$$\overline{\omega}_z=\{z_j=j\tau,\ j=0,1,\ldots,N,\ z_N=L\},$$

where f is the focus of the lens, $Z_R = 0.5k\omega_0{}^2$, ω_0 is the width of the initial signal. An example of the discrete grid is given in fig. 1.

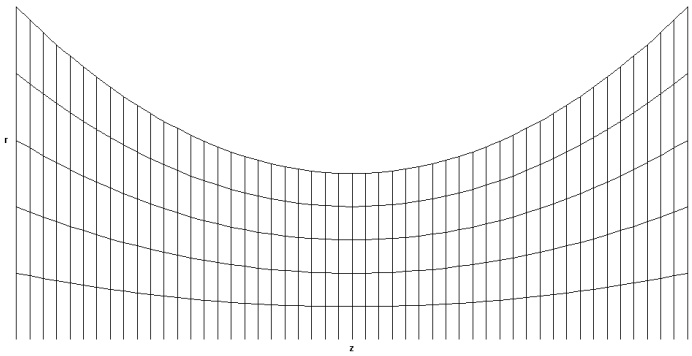

Figure 1. An example of the discrete grid.

We denote the discrete approximates of the functions e_L and e_S by u and v. The following notations are used in our paper:

$$
\begin{aligned}
&u = u\left(z_j, r_k, t_n\right), \quad v = v\left(z_j, r_k, t_n\right), \\
&u(-1) = u\left(z_{j-1}, r_k, t_n\right), \quad v(-1) = v\left(z_{j-1}, r_k, t_n\right), \\
&\widehat{u} = u\left(z_j, r_k, t_{n+1}\right), \quad \widehat{v} = v\left(z_j, r_k, t_{n+1}\right).
\end{aligned}
$$

The function σ_S is approximated by the discrete function σ, which is defined on the grid $\omega_t \times \widetilde{\omega}_z \times \omega_r$:

$$
\widetilde{\omega}_z = \left\{ z_{j-1/2} = (j-0.5)\tau, j = 1, 2, \ldots, N,\ z_N = L \right\}.
$$

2.2. The splitting scheme

We will solve the system of equations (1.1) - (1.5) using the splitting scheme. The differential operator is split up into pieces corresponding to different physical processes.

2.2.1. The first splitting step: diffraction

$$
\frac{\widetilde{u} - u(-1)}{\tau} - i\mu_L \Lambda \frac{\widetilde{u} + u(-1)}{2} + \frac{\alpha_Z}{2} \frac{\widetilde{u} + u(-1)}{2} = 0, \tag{2.1}
$$

$$
\frac{\widehat{v}(-1) - \widehat{u}}{\tau} - i\mu_S \Lambda \frac{\widehat{v}(-1) + \widehat{v}}{2} + \frac{\alpha_S}{2} \frac{\widehat{v}(-1) + \widehat{v}}{2} = 0. \tag{2.2}
$$

The functions u and v are expanded in series

$$u(-1) = \sum_{l=1}^{P} C_l\left(Z_{j-1}, t_n\right) L_l\left(Z_{j-1}, r_k\right),\ r_k \in \omega_r\left(Z_{j-1}\right),$$

where L_l are the Laguerre-Gauss functions. Then we find the coefficients

$$C_l\left(Z_j, t_{n+1}\right) = \frac{1 - 0.5\tau\alpha_Z}{1 + 0.5\tau\alpha_Z} C_l\left(Z_{j-1}, t_n\right),$$
$$\widetilde{u} = \sum_{l=1}^{P} C_l\left(Z_j, t_{n+1}\right) L_l\left(Z_j, r_k\right),\ r_k \in \omega_r\left(Z_j\right).$$

2.2.2. The second splitting step: the nonlinear counter-action
First, we calculate the predictor σ^P of the sound wave:

$$\frac{\sigma^P - \sigma}{\tau} + \alpha\frac{\sigma^P + \sigma}{2} = \Gamma_\sigma \frac{u + u(-1)}{2}\frac{v^* + v^*(-1)}{2}, \tag{2.3}$$

then we approximate the nonlinear counter-action process

$$\begin{cases} \dfrac{u^\circ - \widetilde{u}}{\tau} = -\Gamma_L \dfrac{\sigma^p + \sigma}{2}\dfrac{v^\circ + v}{2}, \\ \dfrac{v^\circ - v}{\tau} = -\Gamma_S \left(\dfrac{\sigma^p + \sigma}{2}\right)^* \dfrac{u^\circ + \widetilde{u}}{2}, \end{cases} \tag{2.4}$$

and finally we compute the corrector of the sound wave:

$$\frac{\widehat{\sigma} - \sigma}{\tau} + a\frac{\widehat{\sigma} + \sigma}{\tau} = \Gamma_\sigma \frac{u^\circ + \widetilde{u}}{2}\frac{v^{\circ *} + v^*}{2}\ . \tag{2.5}$$

2.2.3. The third splitting step: nonlinear self-focusing

$$\widehat{u} = exp\left(i\eta_L\left(|u^\circ|^2 + 2|v^\circ|^2\right)\tau\right) u^\circ, \tag{2.6}$$

$$\widetilde{v} = exp\left(i\eta_S\left(|\widehat{u}|^2 + 2|v^\circ|^2\right)\tau\right) v^\circ. \tag{2.7}$$

Theorem 2.1. *The solution of the finite difference scheme* (2.1) - (2.7) *converges to the solution of the problem* (1.1) - (1.3)*, and the speed of the convergence is estimated as* $O(\tau + h^2)$.

This theorem is proved by using the method from [2]. Extensive results of numerical experiments are given in [1], [3].

3. PARALLEL IMPLEMENTATION OF THE ALGORITHM

In this section we investigate the parallelization of the algorithm (2.1) - (2.7). The machine model assumed in our paper is that of a distributed memory system. The parallel computer is composed of p heterogeneous processors, labeled from 1 through p. Processors communicate through message passing.

Our goal is to decompose computations into tasks and assign these tasks to different processors. The optimization objective for partitioning is to balance the workload among processors and to minimize the interprocess communication costs [4]. We decide to distribute the data along the z dimension.

Two different situations arise when we try to solve the load balancing problem. The first case is considered in this section. During regular phase of implementation of the scheme (2.1) – (2.7) the whole grid $\omega_z \times \omega_r$ is involved in the computation. In data parallel computation the problem domain is decomposed into p sub-domains and these sub-domains are allocated to processors. The processors simultaneously perform the same functions with different data sets. The execution time of processor i is proportional to the amount of grid ω_z points allocated to processor i. Denote the computational power of the processors by a vector $(v_1, v_2, \ldots, v_p)$ and assume that

$$v_1 \le v_2 \le \ldots \le v_p.$$

We use a *static data decomposition*. The data is aligned to different processors once at the beginning of computation. As was stated above we distribute data arrays along the z dimension and keep them local to each processor along the r dimension. The grid ω_z is distributed by using the *block scattering model* and processor i gets the block of N_i points, where N_i is defined as

$$N_i = \lfloor \frac{v_i}{v_1 + v_2 + \ldots + v_p} \rfloor N, \quad i = 2, 3, \ldots, p,$$
$$N_1 = N - (N_2 + N_3 + \ldots + N_p).$$

The sub-domains are connected at their boundaries, hence after the realization of each time step of the algorithm (2.1) – (2.7) processors exchange their overlapping boundary information with nearest–neighbors. Only boundary vectors of $2D$ arrays of discrete functions u and v are included into the messages. These synchronization points divide the computation into phases and the duration of the phase is determined by the heavily loaded processors.

In the case of homogeneous processors, i.e. $v_1 = v_2 = \ldots = v_p$, such a parallelization of the serial code can be implemented with High – Performance – Fortran (HPF) [4], [5].

3.1. Performance of the parallel code

Results of the measurements of performance of the parallel algorithm (2.1) – (2.7) are presented in this section. The simulation was done for $P = 10$,

Table 1.
The speedup and efficiency of the parallel algorithm for $N = 300$.

p	Time T_p	Speed-up $S(p)$	Efficiency $E(p)$
1	334.3	1.00	1.00
2	187.1	1.79	0.89
3	134.7	2.48	0.83
4	113.0	2.96	0.74
5	108.7	3.08	0.62

Table 2.
The speedup and efficiency of the parallel algorithm for $N = 500$.

p	Time T_p	Speed-up $S(p)$	Efficiency $E(p)$
1	961.3	1.00	1.00
2	506.5	1.90	0.95
3	351.1	2.71	0.90
4	297.4	3.23	0.81
5	250.1	3.84	0.75

$M = 150$, $L = 10$, $T = 45$ and two different values of the grid parameter $N = 300$ and 500. We evaluated the relative speedup $S(p)$ and the efficiency $E(p)$:

$$S(p) = \frac{T_1}{T_p}, \quad E(p) = \frac{S(p)}{p},$$

where T_1 is the CPU time on one processor, and T_p is the CPU time using p processors.

Table 1 displays the results for $N = 300$ and table 2 gives the results for $N = 500$. A homogeneous cluster of RS600 workstations with PVM message–passing library was used in our experiments.

From results given in the tables it is clear that the efficiency of the parallel algorithm increases when the value of N increases, as it follows from the scaled-size-problem theoretical analysis.

We also carried out experiments with heterogeneous clusters of workstations. The computational power of one processor was $v_1 = 1/3$, while the other processors had the computational power $v_i = 1$ for $i = 2, \ldots, p$. In table 3 we present CPU times for homogeneous $T_{homog}(p)$ and heterogeneous $T_{heter}(p)$ strategies of data distribution.

It can be seen from table 3 that a good load balance is obtained in the case of the heterogeneous data distribution algorithm. Static mapping is quite effective for computations that have predictable workloads of the sub–tasks.

3.2. Distributed remapping problem

In this section we consider parallelization of the algorithm during initial transition stage, when the computational region is gradually filled from left to right.

Table 3.
The speedup and efficiency of the parallel algorithm for $N = 500$.

p	$T_{homog}(p)$	$T_{heter}(p)$
2	520.1	270.7
3	361.3	161.6
4	304.2	129.7

During this phase the computational workload of the problem increases linearly from time step to time step. Let denote this workload at time moment t_j by

$$comp(t_j) = \gamma j, \quad j = 1, 2, \cdots, N. \tag{3.1}$$

3.2.1. Static task distribution
One processor will spend the time T_1 for solving the whole problem till the time moment t_N:

$$T_1 = \sum_{j=1}^{N} \gamma j = \frac{\gamma(N+1)N}{2}. \tag{3.2}$$

If we statically decompose data area into p equal sub-domains and distribute them to processors, we get the computation time T_p:

$$T_p = \sum_{j=1}^{N/p} \gamma j + \left(N - \frac{N}{p}\right)\frac{\gamma N}{p} = \frac{\gamma N}{p}\left(\frac{2p-1}{2p}N + \frac{1}{2}\right).$$

Then the speedup $S(p)$ of the parallel algorithm is given by

$$S(p) = \frac{(N+1)\,p^2}{(2p-1)N + p}. \tag{3.3}$$

For large N we can estimate the speedup $S(p)$ and the efficiency $E(p)$:

$$S(p) \approx \frac{p^2}{2p-1}, \quad E(p) \approx \frac{p}{2p-1}.$$

We see that $E(p) \to 1/2$ for large p. Now we will find the optimal *static* domain decomposition.

Theorem 3.1. *Let the computational workload of the problem is defined by* (3.1). *Then the optimal block distribution of the grid ω_z is obtained when the grid is decomposed into $p+1$ sub-domains and the processor p gets two last sub-domains.*

Proof. We will study the cases of $p = 2$ and $p = 3$. A general result can be obtained from a similar analysis.

Let assume that we have two processors and decompose the grid ω_z into two sub-domains with K and $N - K$ points, respectively. It is sufficient to study the case $K \leq N/2$ (see fig. 2).

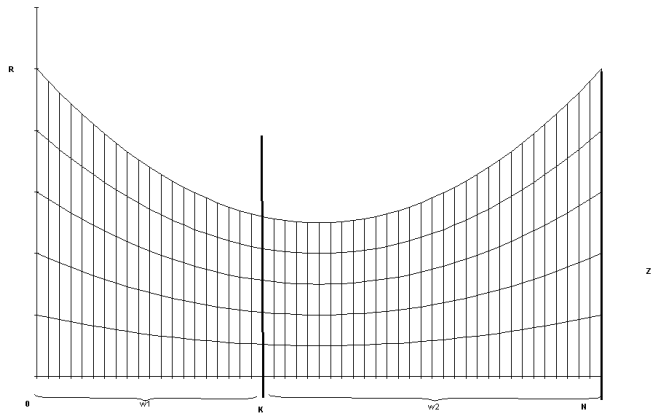

Figure 2. The decomposition of the grid for $p = 2$.

During time steps $1 \leq j \leq K$ only the first processor is involved in computation, during time steps $(K + 1) \leq j \leq 2K$ both processors compute, but the workload of the first processor is larger, and during time steps $(2K+1) \leq j \leq N$ the duration of a phase is determined by the workload of the second processor. Taking into account (3.1) we get the following computation time estimate:

$$\begin{aligned} T_2(K) &= \frac{K(K+1)}{2} + K^2 + \sum_{j=2K+1}^{N} (j - K) \\ &= \frac{3K^2 + K}{2} + \frac{N(N+1)}{2} - K(N+1). \end{aligned}$$

The optimal decomposition parameter K is obtained from the equation

$$T_2'(K) = 3K + \frac{1}{2} - N - 1 = 0,$$

it is given by $K_0 = (N + 0.5)/3$.

Now assume that we have three processors and decompose the grid ω_z into three sub-domains with K_1, K_2 and $N - K_1 - K_2$ points, respectively (see fig. 3). The computation time $T_3(K_1, K_2)$ can be estimated as:

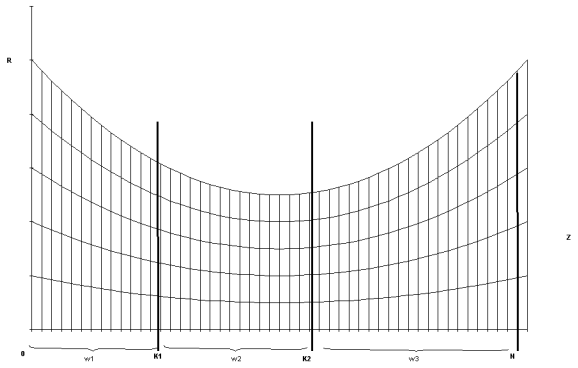

Figure 3. The decomposition of the grid for $p = 3$.

$$T_3(K_1, K_2) = \frac{K_1(K_1+1)}{2} + K_1^2 + \sum_{j=1}^{K_2-K_1}(K_1+j) + K_2^2$$
$$+ \sum_{j=1}^{N-(K_1+2K_2)}(K_2+j) = \frac{3K_1^2+K_1}{2} + (K_2-K_1)\frac{K_1+K_2+1}{2}$$
$$+K_2^2 + \frac{(N-K_1-2K_2)(N-K_1+1)}{2}.$$

Optimal decomposition parameters K_1 and K_2 are obtained from the following system of equations

$$\begin{cases} \frac{\partial T_3}{\partial K_2} = K_1 + 3K_2 - N - \frac{1}{2} = 0, \\ \frac{\partial T_3}{\partial K_1} = 3K_1 + K_2 - N - \frac{1}{2} = 0, \end{cases}$$

and they are given by $K_1 = (N+0.5)/4$, $K_2 = (N+0.5)/4$. Note that a general case of p processors can be investigated by using the symmetry analysis. ■

Table 4 presents the speedup S(p) of the parallel algorithm for the static domain decomposition into p sub-domains and the speedup $S_o(p)$ and the efficiency $E_o(p)$ for the optimal static domain decomposition. It can be seen from the table that the optimal mapping algorithm is important only for a small number of processors.

3.2.2. Data redistribution algorithm

It follows from results given above, that the static domain decomposition is not satisfactory, since computational workloads across the processors are not even

Table 4.
The speedup and efficiency of the static decomposition for $N = 200$.

p	$S(p)$	$S_o(p)$	$E_o(p)$
2	1.336	1.500	0.750
3	1.804	2.000	0.667
4	2.291	2.500	0.625
5	2.784	3.000	0.600

during execution. Hence we will use the *dynamic remapping* algorithm [4]. Dynamic remapping produces better load–balances at the cost of additional data communication overheads. Therefore an application of redistribution algorithm must produce enough benefits that would outweigh the run–time overheads incurred.

The communication costs of sending one message with M columns of data are estimated as:

$$T_{comm} = \alpha + \beta M, \tag{3.4}$$

where α is a startup time, and β is a time required to send one column of data. We have that $\beta \gg \alpha$ for large computation grids.

The optimal redistribution step. In this section we determine the size of the redistribution interval. The computational workload is estimated by model (3.1) and communication costs are given by (3.4).

The following **algorithm** is proposed. We simulate twice the computation time $T_p(r)$ required to implement $2p$ steps of the basic splitting scheme, where $T_p(p)$ denotes the computation time for the parallel algorithm with distributed remapping after p steps and $T_p(2p)$ denotes the time obtained without redistribution. If $T_p(p) \leq T_p(2p)$, then the optimal number of simulation steps between two successive redistribution instances is p. Else we compare $T_p(2p)$ and $T_p(4p)$, i.e. the redistribution algorithm is implemented after $2p$ simulation steps. Such process is iterated till we find that $T_p(lp) \leq T_p(2lp)$ for some l.

We present examples of this analysis. Let consider the case of $p = 2$. The computation times $T_2(2)$ and $T_2(4)$ are given by

$$T_2(2) = 6\gamma + 2\alpha + 2\beta, \quad T_2(4) = 7\gamma + \alpha + 2\beta.$$

Therefore, if $\alpha \leq \gamma$ then the optimal number of simulation steps between two successive redistribution instances is 2. Otherwise we compare $T_2(4)$ and $T_2(8)$, it can be easily verified that

$$T_2(4) = 23\gamma + 2\alpha + 4\beta, \quad T_2(8) = 26\gamma + \alpha + 4\beta.$$

Therefore, if $\alpha \leq 3\gamma$ then the optimal redistribution step is 4. We see that only startup time and computation costs are important in this analysis.

Similarly in the case of $p = 3$ we get that

$$T_3(3) = 9\gamma + 2\alpha + 4\beta, \quad T_3(6) = 11\gamma + \alpha + 4\beta.$$

Dynamic load balancing. In this section we find when the remapping of data should be invoked. Let assume that the optimal redistribution step is p.

We compare two strategies of computation : the *static data distribution* during the whole simulation and dynamic data remapping, when the redistribution step is involved once at time moment t_{N-2p}. If the dynamic remapping is a more efficient strategy, then the same algorithm is applied to the grid ω_z with $N - 2p$ points. The algorithm is iterated till we find the grid for which static decomposition is optimal.

As an example we apply this algorithm to the case $p = 2$. It can be easily verified that

$$T_2(static) = \frac{3N^2 + 4N}{8}\gamma, \quad T_2(dynamic) = \frac{3N^2 - 4N + 8}{8}\gamma + \alpha + \beta.$$

We see that the redistribution of data should be invoked at the end of computation if

$$\alpha + \beta \leq (\frac{N}{2} + 1)\gamma.$$

From this inequality we can determine the number N_0 which defines the starting point of the redistribution strategy:

$$N_0 = \frac{2(\alpha + \beta)}{\gamma} - 1.$$

4. CONCLUSIONS

A splitting finite difference scheme is presented to solve a problem of nonlinear optics. The parallelization of the algorithm is investigated. Static data decomposition is used to decompose the problem domain into p sub-domains. The efficiency of this algorithm is investigated for homogeneous and heterogeneous clusters of workstations. We found that the parallel algorithm is effective if the workloads of processors do not change during the whole computation time.

We also investigated the load balancing problem for an initial stage of computation, when the computation region increases linearly from time step to time step. The optimal static subdivision of the domain is obtained for this problem. Dynamic remapping algorithm is investigated in order to improve the load balancing. Our theoretical model gives a condition when the remapping of data should be invoked and the frequency of redistribution process is determined.

REFERENCES

[1] Raim. Čiegis and A.Dement'ev. Numerical simulation of counteracting of focused laser beams in a nonlinear media. *Mathematical modelling and Applied Mathematics. A. Samarskij and M. Sapagovas (Eds.)*, North-Holland, 1992, 99 – 108.

[2] R. Čiegis. Numerical modelling of the interaction of focused laser beams. *Lith. Math. J.*, **29** (3), 1989, 590 – 607.

[3] V. Girdauskas, A. Dement'ev, G. Kairytė and R. Čiegis. Influence of the beam aberrations and Kerr nonlinearity of medium on the effectiveness and pulse quality of SBS–compressor. *Lith. J. of Physics*, **37** (4), 1997, 319 – 326.

[4] Ch. Xu and F. Lau. *Load Balancing in Parallel Computers: Theory and Practice* . Kluwer Academic Publishers, 1997.

[5] High Performance Fortran Language Specification, 1994.

NETIESINĖS OPTIKOS UŽDAVINIO LYGIAGRETUSIS SKAIČIAVIMO ALGORITMAS

R. ČIEGIS, A. DEMENT'EV, P. RATĖ

Straipsnyje sprendžiama netiesinių diferencialinių lygčių sistema, aprašanti lazerio spindulių sąveiką netiesinėje terpėje. Sudaryta diskrečioji išskaidymo schema bei pateikta lygiagrečioji algoritmo versija. Ištirtas lygiagrečiojo algoritmo efektyvumas. Skaičiavimai atlikti virtualiajame kompiuterių, sujungtų lokaliuoju tinklu, klasteryje. Išsamiai tiriamas algoritmo efektyvumas heterogeniniams lygiagretiesiems kompiuteriams. Duomenų paskirstymui naudojamas statinis paskirstymo metodas. Pateikti ir išanalizuoti duomenų perskirstymo algoritmai, pagerinantys darbo paskirstymo tolygumą pradinėje skaičiavimų fazėje. Teorinėje analizėje atsižvelgiama į duomenų perdavimo kaštus.