Biomedical Simulation of Heat Transfer in a Human Heart

Marjan Šterk* and Roman Trobec

Jožef Stefan Institute, Jamova 39, 1000 Ljubljana, Slovenia

Received May 20, 2005

Theory and practical experiences from numerical simulations of heat transfer in the field of medicine are presented in this paper. The cooling of a human heart during surgery was taken as an illustrative example. All phases of the simulation process are described starting with the construction of an irregularly shaped 3-dimensional model. The mathematical model is based on diffusion and Navier−Stokes equations. The system of partial differential equations is solved by finite difference approximation using an explicit time-stepping scheme to obtain the time evolution of the solution for the complete simulated interval, which is typically 1 h. A typical domain is composed of several million voxels; therefore, the program was parallelized to speed up the simulation. A speed-up of 8.2 was obtained on 16 processors in a Linux cluster.

1. INTRODUCTION

The development of fast computer technologies enables simulation of natural phenomena and experiments that would be ecologically problematic, dangerous to humans, or cost vast amounts of money.[1] Many examples can be found in medicine, where performing in vivo experiments and measurements is often difficult, dangerous, or even impossible,[2] while simulation can provide the only safe and inexpensive insight into physiological processes.

Parallel computers are composed of fast, unified computers connected with fast, dedicated communication links. Today, low-cost parallel computers are available with computing clusters.[3,4] The speed-up is defined as the ratio of the execution time on a single processor to that on a parallel computer. The time of running the application on a parallel computer consists of computation and communication time. By increasing the number of processors the ratio of communication time compared to computation time is usually increased, thus both processor performance and communication time have to be improved in order to improve the overall performance of parallel algorithms. In some problems where a significant amount of global communication is needed, e.g. molecular dynamics,[5,6] optimal performance of intracluster communication is particularly important.

Simulation of human heart cooling during surgery will be described. The human body and the heart have to be cooled appropriately in order to slow their vital functions.[7] To lower metabolic requirements, the body and the heart have to be cooled, e.g. by pumping a cold solution through coronary vessels (cardioplegia). For even better cardiac cooling a method of topical cooling is sometimes used,[8] e.g. submerging the heart in cooling liquid. In vivo temperature measurements are invasive and limited to a few test points, while computer simulation provides improved analysis of various cooling.

The mathematical model of the simulated phenomena is based on partial differential equations (PDE). Solving the mathematical model over a discretized domain gives the values of a certain physical quantity at every grid point for each time interval. The domain, i.e., the human heart, is an irregularly shaped three-dimensional object that has to be discretized, usually by imposing a grid, so that it is suitable for numerical solution. The models of the body organs can be created using the Visual Human Dataset (VHD)[9] or similar data sources, based on 2-D slices.[10] It is thus natural and simple to discretize the problem using finite differences.

Some initial results on the simulation of heart cooling based on diffusion have been reported in refs 11 and 2. In this paper we present results obtained using a more complex mathematical model, which takes into account convection, diffusion, and fluid flow as well as higher spatial resolution. The technique used can also be applied for the prediction of temperature elevation following coronary artery occlusion, cryotherapy simulation,[12] and many other medical applications.

The structure of the paper is as follows. In the next section, a short description of a high resolution 3D-heart model and the formation of the problem domain is described. In section 3, the mathematical background of the simulation and numerical methods are presented. In section 4 the parallelization of the program code is described, and the parallel performances are analyzed. Finally, the obtained results are shown. The paper concludes with a summary of results and some directions for the future work.

2. SPATIAL HEART MODEL

Digital photographs of frozen cross sections of the human thorax from the VHD were used to build a model of the heart. Resolution in the *xy* plane is 0.33×0.33 mm, and consecutive pictures follow each other in the *z* direction at 1 mm intervals. We selected 156 pictures that include parts of the heart from its base to its apex (VHD files from a_vm1350.raw to a_vm1505.raw). The heart area was cropped out of each picture as a square of 512×512 pixels. A marked reference point was found in each picture to ensure precise overlapping, because the original VHD cross sections are not positioned exactly in *x* and *y* directions.

* Corresponding author e-mail: marjan.sterk@ijs.si.

10.1021/ci050206p CCC: $30.25
Published on Web 10/25/2005

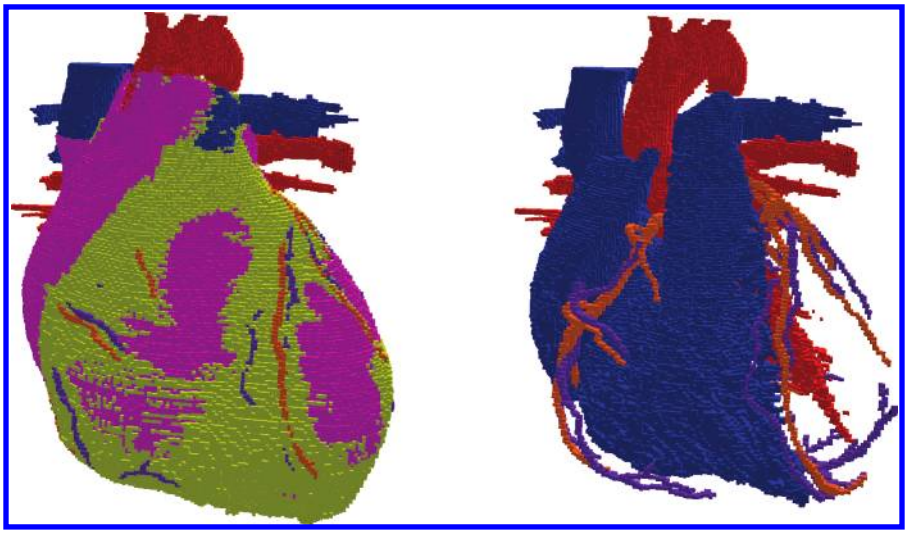

Figure 1. 3-D human heart model in a resolution of 1 mm (left), heart chambers and coronary vessels (right).

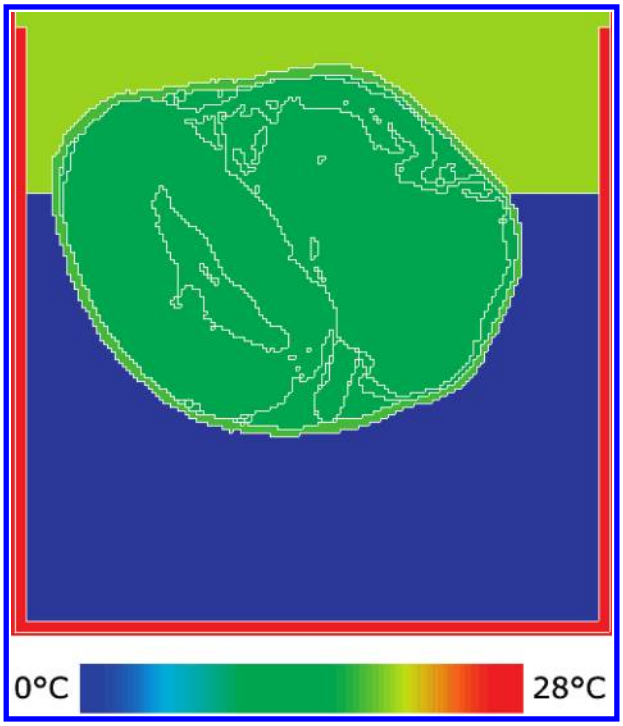

Figure 2. The initial state of heart cooling simulation in slice 109.

Different heart tissues had to be distinguished, because they have different thermodynamic properties. The segmentation of tissues on each slice was done manually using Adobe Photoshop. By putting all the slices together on the *z*-axis, the heart can be visualized as a 3-D model composed of small cubes (voxels). All errors made during 2-D editing can be manually corrected using a spatial editor custom developed for these tasks.

The segmented slices were resized to a resolution similar to that in the *z* direction. The final 3-D model used in the simulation is shown in the left part of Figure 1. The right part shows heart chambers and coronary vessels only. Myocardium, pericardium, and fat are omitted for better visibility.

2.1. Problem Domain. The conditions during cardiac surgery are imitated with an isolated cube large enough to contain the simulated heart model, as shown in Figure 2. The lower part of the cube represents a cooling container and is filled with a cooling liquid, which is initially still at a temperature of 0.2 °C. The container walls are kept at a constant temperature of 28 °C to simulate moderate body hypothermia. The upper part of the cube represents the open thorax surrounded by air at room temperature, which due to circulation has a constant temperature of 20 °C. The cube is composed of 163 × 169 × 163 voxels of 1 mm^3 and independently characterized by specific thermodynamic constants, which were taken from ref 13. We neglected anisotropy because no adequate thermal constants for tissues were available in the literature and because tissue cells are much smaller than spatial resolution used.

The heart is supposed to have been previously cooled by cardioplegia, so the initial temperatures of its tissues and cavities are taken to be 12.7 °C. We simulated cardioplegia and topical cooling, which in our case consists of substituting the partially warmed cooling liquid with fresh liquid at 0.2 °C every 5 min. The temperature gradients inside the heart are small, allowing us to approximate the cavities as solid tissue.

3. MATHEMATICAL MODEL AND NUMERICAL SOLUTION

The heart cooling in the problem domain was modeled by a set of coupled partial differential equations (PDEs) that describe the temperature and the velocity in the domain of interest as a function of place and time. These PDEs have to be discretized to give a system of algebraic equations, which is then solved numerically.

3.1. Governing Equations. The basic equation that governs the heat transfer is known as the *heat conduction equation*, which is derived for example in ref 14. For isotropic substances it can be expressed as

$$\frac{\partial T}{\partial t} = \frac{1}{\rho c_p}\nabla\cdot(\lambda\nabla T) - (\mathbf{v}\cdot\nabla)T \qquad (1)$$

where T is the temperature, ρ, c_p, and λ are thermodynamic constants, and $\mathbf{v}$ is the velocity. Fluid flow was modeled by the *Navier−Stokes equation*, described for example in refs 15 and 16

$$\frac{\partial \mathbf{v}}{\partial t} = \frac{\eta}{\rho}\nabla^2\mathbf{v} - \frac{\nabla p}{\rho} - (\mathbf{v}\cdot\nabla)\mathbf{v} + \mathbf{a} \qquad (2)$$

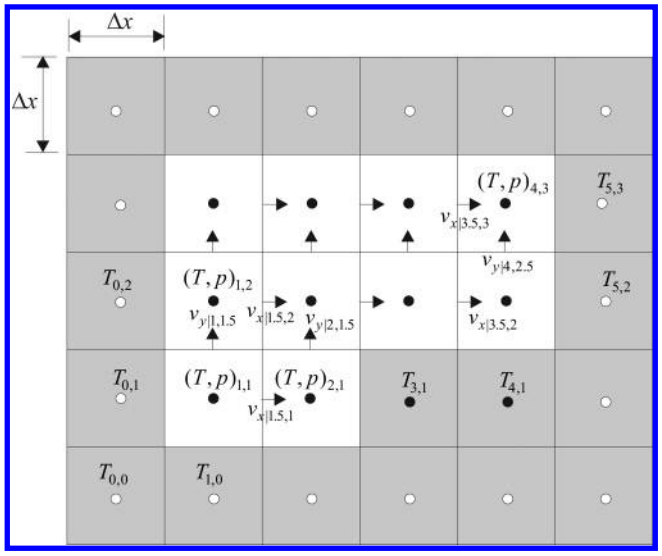

Figure 3. Staggered temperature/pressure and velocity grid. The solid voxels are shown in gray, while the constant temperatures on the global domain boundary are represented by white dots.

where **a** is the buoyancy acceleration resulting from temperature dependence of the density. It was expressed as

$$\mathbf{a} = \mathbf{g}\hat{\rho}(T),\ \hat{\rho}(T) = \frac{\rho(T) - \rho(T_0)}{\rho(T_0)} \tag{3}$$

where **g** is the gravitational acceleration and T_0 is the reference temperature 0 °C. The *continuity equation* for a mass flow has to be valid in the whole simulated domain.[15] For an incompressible fluid it is expressed as

$$\nabla\cdot\mathbf{v} = 0 \tag{4}$$

Initial values are needed for **v**, T, and p. Boundary conditions for a fixed wall prescribe that the velocity and the normal component of pressure gradient are zero, while temperature boundary conditions are given by the constant temperature of container walls. Transient conditions for boundaries between two substances, e.g. water and air, require continuous temperature field.[14,15,17] Transient conditions for velocity and pressure are not needed because there are no inside boundaries between two liquid substances.

3.2. Discretization. The problem domain was discretized to voxels that can be either liquid or solid. For solid voxels only (1) is used because $\mathbf{v} = 0$ and p is not needed. All boundary voxels are solid to form a closed cavity. A staggered grid is used with T and p defined in the middle of each voxel, while each velocity component is defined on the respective perpendicular voxel edge (see Figure 3 for a 2-dimensional example).

Time Discretization. The time is discretized into time-steps Δt. To couple (4) with (2) the Hirt and Cook scheme[15,18] and explicit Euler time integration were used. The scheme introduces the notion of pressure correction p_Δ, which is obtained by solving the Poisson PDE

$$\nabla^2 p_\Delta = \frac{\rho}{\Delta t}\nabla\cdot\mathbf{v}^* \tag{5}$$

The calculation scheme is shown in Figure 4. It is of second

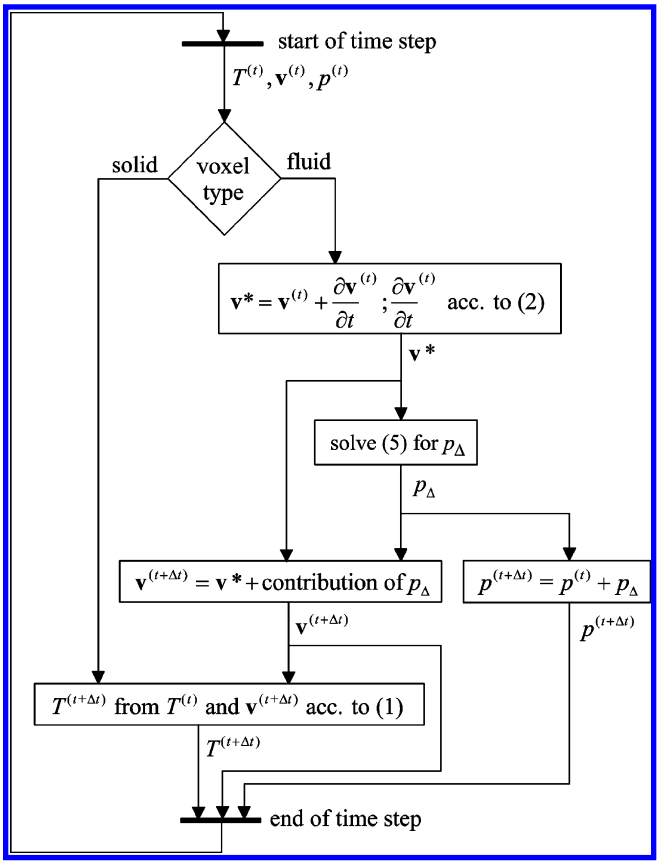

Figure 4. Calculations and data dependencies in every time-step of the simulation.

order accuracy in space and first order in time and is conditionally stable. To ensure stability, the time-step must satisfy

$$\Delta t < \frac{\Delta x}{2v_{\max}},\quad \Delta t < \frac{\rho\Delta x^2}{6\eta},\quad \Delta t < \frac{\rho c_p \Delta x^2}{6\lambda} \tag{6}$$

where $v_{\max}$ is the maximum velocity in the domain within the time-step. The factor 2 in the first condition is a consequence of the staggered grid used, i.e., the domain of dependence in each time-step must be smaller than half the grid spacing. Additionally, Δt must be small enough to ensure accuracy. In our case, the time-steps 50 times and 2 times smaller than suggested by (6) yielded temperatures differing by less than 0.1 °C, so the latter was used. Longer time-steps were also tested but yielded significantly lower accuracy.

Finite Difference Schemes. The finite difference schemes for the diffusion equation and discontinuous thermodynamic constants on substance boundaries are described in detail in refs 17 and 19. The forward difference formula is used for all time derivatives because explicit Euler's method is used for integration.[20] The staggered grid enables the use of central differences for most spatial derivatives. However, some terms require careful analysis. For example, using the central difference for the term $(\mathbf{v}\cdot\nabla)\mathbf{v}$ of the Navier−Stokes equation introduces oscillations into the solution because the velocity correction at odd points only depends on the velocities at even points and vice versa. Instead, we use the upwind formula[21] shown graphically in Figure 5.

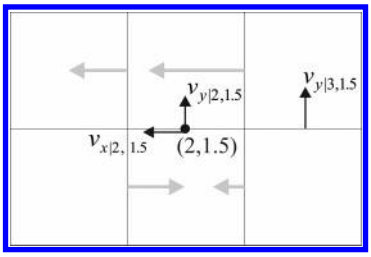

Figure 5. To calculate $(\partial v_y/\partial x)$ at the black point (2,1.5), v_x at that point is first interpolated from the neighboring v_x values (shown in gray). The value obtained is negative (the fluid flows to the left); therefore, the central and right values of v_y are used in the upwind formula (7).

$$\left.\frac{\partial v_y}{\partial x}\right|_{x,y+(1/2),z} = \begin{cases} \dfrac{v_{y|x+1,y+(1/2),z} - v_{y|x,y+(1/2),z}}{\Delta x} & v_{x|x,y+(1/2),z} < 0, \\ \dfrac{v_{y|x,y+(1/2),z} - v_{y|x-1,y+(1/2),z}}{\Delta x} & v_{x|x,y+(1/2),z} \geq 0, \end{cases} \tag{7}$$

Special schemes have to be used at boundaries for some derivatives, for example $\partial^2 v_x/\partial y^2|_{(3.5,3)}$ in Figure 3. The boundary condition $v_{x|3.5,3.5} = 0$ is approximated by setting $v_{x|3.5,4} = -v_{x|3.5,3}$. The formula for the approximation of $(\partial^2 v_x/\partial y^2)$ at the mesh point $(x + (1/2), y, z)$ with a solid cube above it is thus

$$\left.\frac{\partial^2 v_x}{\partial y^2}\right|_{x+(1/2),y,z} = \frac{v_{x|x+(1/2),y+1,z} - 2v_{x|x+(1/2),y,z} + v_{x|x+(1/2),y-1,z}}{\Delta x^2} = \frac{-3v_{x|x+(1/2),y,z} + v_{x|x+(1/2),y-1,z}}{\Delta x^2} \tag{8}$$

Obtaining Pressure Correction. Solving (5) becomes the most computationally intensive part of the simulation. To enforce the pressure boundary conditions, Neumann boundary conditions are prescribed for the pressure correction,[15] i.e., the normal derivative

$$\frac{\partial p_\Delta}{\partial n} = 0 \tag{9}$$

on all boundaries. There are infinitely many solutions that satisfy (5) and (9), differing only in an additive constant. Because the absolute values of pressure are not important in this kind of problems, any of the particular solutions suffices. The Poisson eq 5 is also discretized using finite differences, giving a system of linear equations $A\mathbf{p}_\Delta = \mathbf{b}$.

The system matrix A contains 7 nonzero elements in each row, which makes iterative solvers the obvious choice. We tested SOR, preconditioned conjugate gradient, and full multigrid method. As expected, the full multigrid solves the system in a constant number of iterations regardless of system size and is thus the most efficient. A single iteration of the full multigrid method is composed of multiple transitions between grids of different resolutions and running a few iterations of the Gauss-Seidel method on each. Details on multigrid implementation on irregular domains can be found in ref 22.

4. PARALLEL IMPLEMENTATION

The simulation was parallelized using one-dimensional domain decomposition in order to preserve simplicity and to enable effective execution on all clusters that can embed at least the ring topology. Each processor runs the simulation on its subdomain, i.e., a certain number of consecutive slices. To implement calculations from Figure 4 in every point of the subdomain, the values of T, $\mathbf{v}$, and p in the neighboring slices are needed. Because the latter are updated in every calculation, point-to-point communication is required afterward, i.e., each data dependency in Figure 4 implies that neighboring processors exchange the values from the subdomain edge slices.

The multigrid method is parallelized following the same principles. Computations on fine grids are done in parallel using the same domain decomposition, while for the coarsest grids data are gathered on a single processor for sequential computation.[23] The total computational effort on the coarsest grids is negligible compared to the finest grid. Global communication is needed at the end of each iteration for the implementation of stopping criteria.

Following the above principles, the computation time would scale as t_s/P, where t_s is the time of sequential execution and P is the number of processors, however, only in the case of ideal load balancing. It is quite complicated to automatically balance the computation load for general domains because solid as well as fluid voxels would have to be evenly distributed among processors. The computational complexity for fluid voxels is approximately 35 times greater than for solid voxels. Automatic load balancing was not implemented in this stage; therefore, limited speed-up is expected.

The time for point-to-point communication (subdomain edge slices) is proportional to $n^{2/3}$, where n is the number of voxels, and independent of P. The time for global communication (gathering the coarse grid data on a single processor and stopping criteria) is proportional to n_{seq} and P, respectively, where n_{seq} is the number of grid points on the coarsest grid that is computed in parallel. The speed-up will vary depending on the load balancing and the ratio between calculation and communication time. Because the communication time does not decrease with larger P and due to unbalanced computation, we cannot expect any significant speedup above some number of processors.

5. RESULTS

Before the simulation of the heart cooling, we performed various standard tests e.g. driven cavity. Results were in accordance with expectations, hence we concluded that our simulation program will give correct results also in cases simulated in this paper.

Simulation results for the temperature and velocity fields after 30 min in slice 109 are given in Figure 6. The temperature ranges are the same as in Figure 2. The velocity field is represented with arrow lengths (box width/15 = 1 mm/s).

To validate the simulation results we compared them with measurements of 15 porcine hearts of similar dimensions

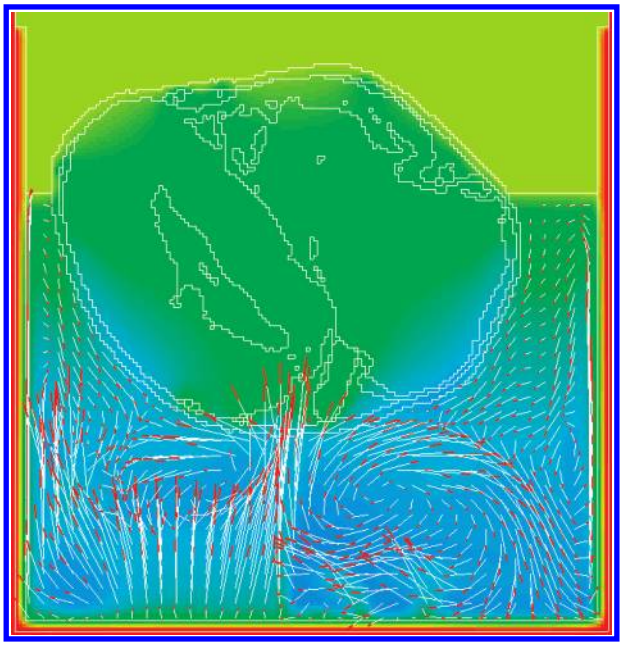

Figure 6. The simulated temperature and velocity fields after 33 min in slice 109.

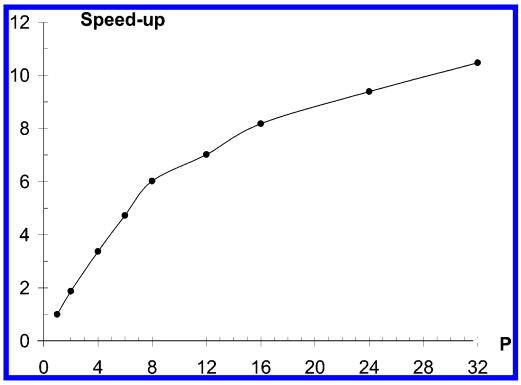

Figure 7. Speed-up of the parallel simulation program.

and in the same conditions as in the simulation example.[24] Quantitative comparison was not possible because of the differences in hearts dimensions, unreliable measurements particularly with the surface probe, and not exactly maintained simulated conditions, for example, changing the cooling liquid in 5 min intervals. We can just recognize qualitative similarity in temperature trends. More precise validation of the simulated results is a topic of a separate paper.

The parallel simulation program was run on a computing cluster built of 16 processing nodes connected in a toroidal 4-mesh. Each node contains two 64-bit processors (AMD Opteron 244), 1024 MB RAM, and six Gigabit Ethernet ports (2 × Broadcom BCM5704C + 4 × Intel Pro/1000 MT). Besides the four mesh neighbors, all nodes are also directly connected to a Gigabit switch (Level One GSW-2451T). The software used includes Fedora Core 2 Linux with kernel 2.6.8−1.521smp, LAM/MPI v3.0.7. communication library,[25] and gcc 3.3.3.

The sequential execution time required to simulate 1 s was equal to 370.9 s. Obtained speed-up for larger numbers of processors is shown in Figure 7. As expected, the speed-up is close to linear for the number of processors less than or equal to 8 and significantly lower than linear as the number of processors grows. As noted, this behavior is a consequence of an increasing proportion of time spent for communication and of load imbalance. The first problem could be partially alleviated by using 2-D or 3-D domain decomposition. The second problem is much harder to solve adequately, because any regular decomposition that divides liquid points evenly will divide solid points unevenly and vice versa.

6. CONCLUSIONS

In this paper it was shown that parallel computer simulation can be used to predict temperature distribution at different critical points of a heart. In addition, new cooling methods and certain other medical procedures can also be evaluated by similar approaches. We are aware that the described simulation has several limitations regarding realistic treatment of boundary conditions and other external influences in the operating room. However, it was demonstrated that real-time medical simulations can be expected soon. They will serve either to train the personnel or for the analysis of different methods applicable for personalized treatment of individual patients.

One of the most important problems in numerical computation still remains the solution of large sparse linear equation systems. Our results show that the multigrid method outperforms other linear solvers because the number of iterations is independent of the domain size. The method is scalable well, and therefore parallel computers can be beneficially used throughout the simulation.

The sequential version of the simulation is impractical because it runs more than 350 times slower than in the real time. The parallel version achieved speedup of 8.2 on 16 processors and 10.5 on 32 processors. The possibilites to linearly scale the simulation to such a processor number include improving load balancing and lowering communication requirements by employing two-dimensional domain decomposition.

In medical simulations, moving domains are typically encountered, therefore as the most advanced solution mesh-free methods[26] should be considered. Although these methods are more complex by their formulation and more computationally demanding, they could be beneficial, particularly if running on parallel computers.

ACKNOWLEDGMENT

This research was funded under Grant No. P2-095-0106 by the Ministry of Higher Education, Science and Technology of Slovenia.

REFERENCES AND NOTES

(1) Martino, R. L.; Johnson, C. A.; Suh, E. B.; others. Parallel computing in biomedical-research. *Science* **1994**, *265*, 902−908.
(2) Trunk, P.; Gersak, B.; Trobec, R. Topical cardiac cooling − computer simulation of myocardial temperature changes. *Comput. Biol. Med.* **2003**, *33*, 203−214.
(3) Braga, A. A. C. Technical aspects of beowulf cluster construction. *Quim. Nova* **2003**, *26*, 401−406.
(4) Borštnik, U.; Hodošček, M.; Janežič, D. Improving the performance of molecular dynamics simulations on parallel clusters. *J. Chem. Inf. Comput. Sci.* **2004**, *44*, 359−364.
(5) Janežič, D.; Praprotnik, M. Molecular dynamics integration time step dependence of the split integration symplectic method on system density. *J. Chem. Inf. Comput. Sci.* **2003**, *43*, 1922−1927.
(6) Trobec, R.; Šterk, M.; Praprotnik, M.; Janežič, D. Parallel programming library for molecular dynamics simulations. *Int. J. Quantum Chem.* **2004**, *96*, 530−536.

(7) Karthik, S.; Grayson, A. D.; Oo, A. Y.; others. A survey of current myocardial protection practices during coronary artery bypass grafting. *Ann. R. Coll. Surg.* **2004**, *86*, 413−415.
(8) Olin, C. L.; Huljebrant, I. E. Topical cooling of the heart − a valuable adjunct to cold cardioplegia. *Scand. J. Thorac. Card.* **1993**, *41*, 55−58.
(9) Ackerman, M. J. The visible human project. *Proc. IEEE* **1998**, *86*, 504−511.
(10) Trobec, R.; Pipan, G.; Trunk, P.; Močnik, J. Spatial heart model derived from VHD. In *Bioimages for Europe '99, 2nd International Workshop of the Visible Human Dataset*; Milan, 1999.
(11) Trobec, R.; Slivnik, B.; Geršak, B.; Gabrijelčič, T. Computer simulation and spatial modelling in heart surgery. *Comput. Biol. Med.* **1998**, *4*, 393−403.
(12) Salcido, R.; Musick, D. W.; Erdman, F. The erdman therapy − a treatment utilizing hot and cold therapy. *Am. J. Phys. Med. Rehab.* **2003**, *82*, 972−978.
(13) Bowman, H. F.; Cravalho, E. G.; Woods, M. Theory, measurement, and application of thermal properties of biomaterials. *Annu. Rev. Biophys.-Bioeng.* **1975**, *4*, 43−80.
(14) Özisik, M. N. *Finite Difference Methods in Heat Transfer*; CRC Press: Boca Raton, 1994.
(15) Fletcher, C. A. J. *Computational Techniques for Fluid Dynamics*; Springer-Verlag: 1988.
(16) Kuščer, I.; Kodre, A. *Mathematik in Physik und Technik*; Springer-Verlag: Berlin, 1993.
(17) Praprotnik, M.; Šterk, M.; Trobec, R. Inhomogeneous heat-conduction problems solved by a new explicit finite difference scheme. *Int. J. Pure Appl. Math.* **2004**, *13*, 275−291.
(18) Hirt, C. W.; Cook, J. L. Calculating three-dimensional flows around structures. *J. Comput. Phys.* **1972**, *10*, 324−340.
(19) Šterk, M.; Trobec, R.; Praprotnik, M. Numerical schemes for fluid flow and heat transfer in medical simulations. *Parallel Distributed Comput. Pract.* **2002**, *5*, 321−329.
(20) Golub, G.; Ortega, J. M. *Scientific Computing − An Introduction with Parallel Computing*; Academic Press Inc.: Boston, 1993.
(21) Heath, M. T. *Scientific Computing: An Introductory Survey*, 2nd ed.; WCB/McGraw-Hill: 2002.
(22) Šterk, M.; Trobec, R. A multigrid poisson solver on general 3-dimensional domains. In *Parallel processing and applied mathematics: 5th International Conference (Lecture notes in computer science, 3019)*; Springer: 2003.
(23) Šterk, M.; Trobec, R. Parallel performance of a multigrid poisson solver. In *Second International Symposium on Parallel and Distributed Computing ISPDC 2003: proceedings*; IEEE Computer Society: 2003.
(24) Trunk, P.; Trobec, R.; Gersak, B. Measurement of porcine heart temperatures. *Pflgers Arch.* **2000**, *440*, R132−R133.
(25) Snir, M.; Otto, S.; Huss-Lederman, S.; Walker, D.; Dongarra, J. *MPI: The Complete Reference*; The MIT Press: 1996.
(26) Liu, G. R. *Mesh Free Methods: Moving beyond the Finite Element Method*; CRC Press: Boca Raton, 2003.

CI050206P