LA-UR- 09-00386

Approved for public release;
distribution is unlimited.

Title: A HYBRID PARALLEL FRAMEWORK FOR THE CELLULAR POTTS MODEL SIMULATIONS

Author(s): KEJING HE
YI JIANG
SHOUBIN DONG

Intended for: PROCEEDINGS/MEETING
CC GRID 2009
IEEE INTERNATIONAL SYMPOSIUM ON CLUSTER COMPUTING AND GRID

Form 836 (7/06)

A Hybrid Parallel Framework for the Cellular Potts Model Simulations

Kejing He*, *Member, IEEE,* Yi Jiang† and Shoubin Dong*
*School of Computer Science and Engineering, South China University of Technology, Guangzhou, China 510641
†Theoretical Division, Los Alamos National Laboratory, Los Alamos, New Mexico 87544
Email: kejinghe@ieee.org, jiang@lanl.gov, sbdong@scut.edu.cn

***Abstract*—The Cellular Potts Model (CPM) has been widely used for biological simulations. However, most current implementations are either sequential or approximated, which can't be used for large scale complex 3D simulation. In this paper we present a hybrid parallel framework for CPM simulations. The time-consuming PDE solving, cell division, and cell reaction operation are distributed to clusters using the Message Passing Interface (MPI). The Monte Carlo lattice update is parallelized on shared-memory SMP system using OpenMP. Because the Monte Carlo lattice update is much faster than the PDE solving and SMP systems are more and more common, this hybrid approach achieves good performance and high accuracy at the same time. Based on the parallel Cellular Potts Model, we studies the avascular tumor growth using a multiscale model. The application and performance analysis show that the hybrid parallel framework is quite efficient. The hybrid parallel CPM can be used for the large scale simulation ($\sim 10^8$ sites) of complex collective behavior of numerous cells ($\sim 10^6$).**

I. Introduction

The Cellular Potts Model (CPM) [1] is a simple and flexible framework for cell-oriented models of development. It has been used for modeling many morphogenesis processes, including cell sorting [2], tumor growth [3], vascularization [4], angiogenesis [5], limb growth [6] and slime mold development [7]. The underlying of the CPM is a three-dimensional rectangular lattice. Every site of the lattice has a value (formally ID), and the sites with the same ID form a cell, which thus has a volume and shape. In this approach, cells are deformable and can interact with each other through the cell boundaries. The cells can have *Type* and *State* as part of their properties. The cell-cell interactions are described through an effective total energy (formally *Hamiltonian*), which typically include a cell-type dependent surface adhesion energy and a volume energy:

$$\begin{aligned}\mathcal{H} &= \sum_{(i,j)} J_{\tau(\sigma_i),\tau(\sigma_j)}[-\delta(\sigma_i,\sigma_j)] \\ &\quad +\lambda_{volume}\sum_{\sigma}[V(\sigma)-V_{target}(\sigma)]^2 \\ &\quad +\lambda_{surface}\sum_{\sigma}[S(\sigma)-S_{target}(\sigma)]^2\end{aligned} \tag{1}$$

where $J_{\tau(\sigma_i),\tau(\sigma_j)}$ is the adhesive energy between ID σ_i's cell type and ID σ_j's cell type. λ_{volume} and $\lambda_{surface}$ are the volume elasticity and membrane elasticity respectively. In general, the second and third terms make the cells grow to their target size gradually. The classic CPM employs a modified Metropolis Monte Carlo algorithm [8] to evolve the system. In the Metropolis algorithm, a random lattice site is chosen, and the ID of this site is updated to be one of the other IDs. Let $\Delta\mathcal{H}$ be the change of energy due to this update, the Metropolis-Boltzmann probability for accepting such update is:

$$P = \begin{cases} 1 & \Delta\mathcal{H} \le 0 \\ \exp(-\Delta\mathcal{H}/T) & \Delta\mathcal{H} > 0 \end{cases} \tag{2}$$

Since the underlying structure of the CPM is a lattice, and the typical discretization scale is several microns per lattice site, a realistic biological tissue level simulation means a large lattice requiring large memory and computing resources. In the case of tumor growth modeling (III-A), the simulation domain consists of up to 400^3 lattice sites, and we need to simulate the tumor growth for more than 30 days. For such simulations, parallelization becomes a necessity.

Domain decomposition is widely used in parallelization for lattice based models. The existing domain-decomposition methods usually decompose a large domain into a number of smaller subdomains, each is designated to a separate physical processor. In addition to the local subdomain information, each processor also has to maintain some data (ghost zones) from its neighbouring subdomains. After one calculation step, the data in the ghost zone are updated, so each processors have current information for its neighboring subdomains. For most lattice based models (such as Cellular Autonoma), domain decomposition method is efficient in parallelization. But because the CPM has a two layer structure (section II-A and Fig. 1), and the Monte Carlo operation (section II-D) in the CPM needs the consistency of the information on cell properties, e.g. cell volume, the parallelization of Monte Carlo lattice update in distributed-memory computing system is difficult in order to achieve accuracy and performance at the same time.

The basis of the CPM, the Potts model, is effectively a generalized Ising model, which has been the example system of parallelization implementation. Barkema et al. [9] found that to reproduce the results obtained from the sequential algorithm, the number of interprocess synchronizations during one Monte Carlo step (MCS) should be statistically equal to N_s, where N_s is the number of sites located on the border of adjacent subdomains. They also presented a "smart" algorithm that reduces the synchronization number to $\sqrt{2N_s/\pi}$ in two-dimensions (2D). According to this estimate, for a 400^3 lattice, the processes have to synchronize tens of thousands of times

each MCS, which would severely limit the parallel performance in the distributed memory environment. To reduce the number of synchronizations, a few methods have been proposed to exchange some accuracy for parallel efficiency. Wright et al. [10] used a "checkerboard" strategy to parallelize the Potts Model. In their approach, each subdomain is divided into four subgrids (in 2D, and eight in 3D) colored with different colors using the same schema. During the "subgrid" step, the processors work on subgrids of one color, and other subgrids are left intact. After each "subgrid" step, the processors are synchronized. Next, the processors work on subgrids of another color. The "checkerboard" approach goes through lattice sites differently than the sequential algorithm. The parallel algorithm does not pick sites randomly, even within a subdomain within a processor [10]. Chen et al. [11] followed the "checkerboard" approach to parallel the Cellular Potts Model, and they analyzed the relationship between the subgrid switching frequency, the accuracy, and the efficiency. Cercato et al. [12] developed a parallel algorithm for the CPM based on a Random Walker (RW) approach [13]. All of these approximation algorithm sacrifice some degree of accuracy for efficiency. The full analysis of Monte Carlo lattice update is presented in section II-D.

The development of domain decomposition methods is closely related to the progress of high performance computing. Because the symmetric multiprocessing (SMP) clusters are becoming increasingly common in the top computer systems, it has become desirable for the domain decomposition methods to take advantage of the memory architecture of modern distributed computing systems. An important property of a good parallel domain decomposition method is that it can integrate with both the scientific problem and the underlying computing system architecture. The Message Passing Interface (MPI) [14] is a *de facto* computer communications standard for communication among the nodes running a parallel program on a distributed memory system. Since MPI has been implemented for almost every distributed memory architecture and each implementation is optimized for the hardware on which it runs, MPI is portable and efficient. OpenMP [15] is another well-known application programming interface (API) which supports multi-platform, shared-memory, multiprocessing programming in C/C++ and Fortran on many architectures. It consists of a set of compiler directives, library routines, and environment variables that influence run-time behavior. To achieve both high efficiency and high accuracy, we adopt a hybrid approach, where the time-consuming PDE solving, cell division, and cell death operation are distributed to the SMP cluster, the inter-nodes communication is fast and trivial. Since the parallelization of Monte Carlo lattice update , the Monte Carlo lattice update is parallelized on shared-memory system, through OpenMP. Because the Monte Carlo lattice update is relatively fast than PDE solving, the OpenMP based parallelization is a good balance between performance and accuracy. Section II-A briefly describes the CPM. Section II discusses the parallelization of various CPM operations, including the Monte Carlo lattice update, the advection-diffusion partial differential equation, cell division, and cell death. Why the hybrid approach is essential and advantageous is also analyzed there. The accuracy, performance and productivity of the parallel CPM is demonstrated in section III through a tumor growth problem.

II. Hybrid Parallel Framework

A. Cellular Potts Model

The underlying of the CPM is a three-dimensional lattice. A "cell" σ is a simply-connected domain of lattice sites with the same ID number σ, which could represent a biological cell or a generalized domain entity, such as extracellular matrix. Cell attributes include the cell type (tumor vs. endothelial), cell state (proliferating vs. dead), center of mass, volume, target volume, etc.. The hierarchical structure of the CPM is illustrated in Fig. 1.

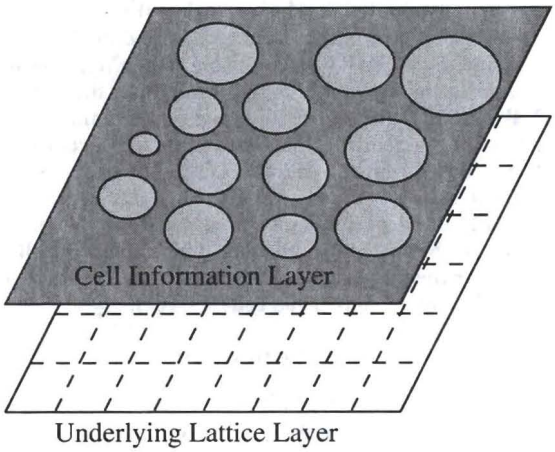


Fig. 1. The hierarchical structure of the Cellular Potts Model

B. Domain Decomposition Method (DDM)

Domain decomposition method generally refers to the splitting of problem into coupled problems on smaller subdomains, which are partitions of the original domain. From section II-A, we see a classic CPM is composed of two layers. The underlying is a regular rectangular lattice, each site in the lattice contains a number identifying which cell this site belonged to. In the computer implementation of the model, the sites in the lattice would be represented by one array, whose indexes correspond to a structure of locations in the lattice. The top layer is not a lattice, but a collection of different cells. If every site interacts with every other site, it is generally hard to divide the computing tasks into a set of parallel procssors. Fortunately, all the operations in the CPM have their local effective region. This means that to apply one specific operation $\circ$ to the system (at the site S) from the time t to the time $t + \triangle t$, we need only the information of the values at time t in the limited surroundings of S. In this paper, this set of information is defined as the *knowledge zone* of site S regarding operation $\circ$. In PCPM, different operations have different kinds of knowledge zones, they are detailed

in section II-D. A natural way to address the parallelization in the CPM is the domain decomposition method: we divide the physical domain into several subdomains (one for each processor); each processor can then model the subsystem, storing only the data associated with the subdomain, and exchanging knowledge zone with the processors responsible for the neighboring subdomains. In PCPM, the data need to be exchanged may include the underlying sites' ID numbers and the cell information associated with these IDs.

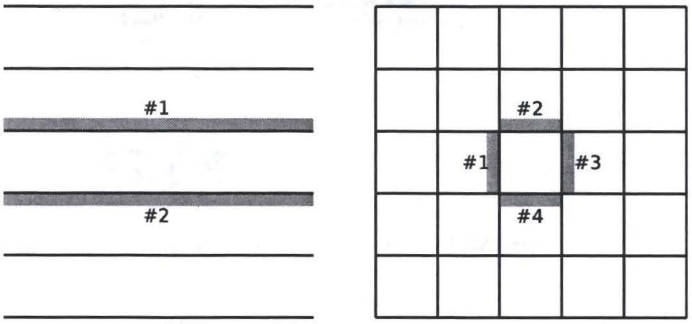


Fig. 2. The different communication patterns in 1D- decomposition and 2D-decomposition. In 1D- decomposition, every ghost zone is continuous. In 2D-decomposition, either ghost zone #1 and #3 or ghost zone #2 and #4 are non-continuous

We choose a 1D-decomposition scheme in PCPM, based on two considerations. First, the communication overhead will influence the performance of PCPM. In a 2D-decomposition in 2D domain, as illustrated in figure 2, each subdomain has four ghost zones, numbered $\#1$ to $\#4$ respectively. Among those four ghost zones, there are two are memory-discontinuous. In MPI, though discrete scatter and gather is possible, the performance of continuous send and receive is much better than discrete scatter and gather. So the memory recomposition is necessary before every neighbour communication. If we use 1D-decomposition, the ghost memory is a continuous bulk, we do not need to recomposite memory before every communication. Second, the CPM is composed of two layers (section II-A). In the 2D- or 3D- decomposition, synchronize two layers between different subdomains and implement all the CPM operations (section II-D) accurately are very difficult, if not impossible. Especially because of the complexity of operations in the CPM, complex communication patterns will decrease the performance of the parallel code. The parallel implement of these operations is detailed in section II-D, the difficulties in implementing these operations acurately in 2D- or 3D- decomposition are also discussed there.

C. Ghost zones

The width of ghost zone is related to the communication data size, and to the performance of parallel programs. In general, the narrower the ghost zone, the better the performance. To facilitate the analysis of required ghost zone width, we introduced the concept of *Safe Cell Width (SCW)*. Every cell in the CPM has a target volume. Most cells do not have a spherical shape. Safe Cell Width (SCW) is a width large enough such that every cell, regardless of its shape, can be contained into a SCW^3 cube. In the parallel CPM, the width of ghost zone is set to be SCW. A typical value of SCW may be twice the target radius ($\sqrt[3]{\frac{3\times(target\ volume)}{4\pi}}$). If some

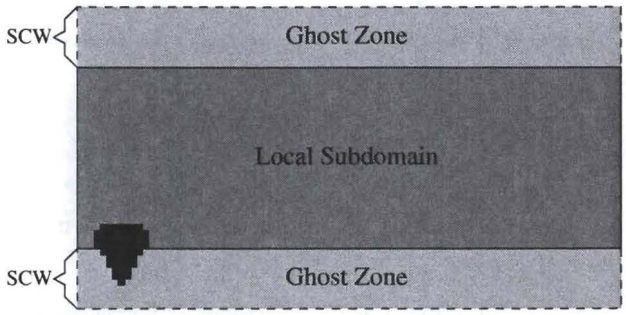


Fig. 3. The local subdomain and ghost zone of domain decomposition method. The width of the ghost zone is set to be Safe Cell Width (SCW).

subdomain γ's local area (dark area in Fig. 3) contains a site whose ID number is σ, the process responsible for subdomain γ is guaranteed to have all the information of cell σ. That is, there is enough information for the parallel operations presented in section II-D.

D. Parallel Cellular Potts Model

A CPM-based simulation is usually composed of several steps or operations. In the case of tumor growth modeling, the operations include the Monte Carlo lattice update, the equation solving, cell growth, division, and cell death. Based on these operations' intrinsic properties, they are parallelized using the shared memory method or the distributed memory method accordingly.

1) Monte Carlo lattice update: As stated in section I, the change of effective energy $\triangle E$ is related to cells' current volume. And if the update is accepted, cells' current volume should also be updated. In the domain decomposition method, the hierarchical structure of the CPM posed a latent performance drawback. As illustrated in Fig. 4, there is a cell σ on the boundary of two subdomains. In one specific Monte Carlo operation, both of the subdomain I and subdomain II have σ's lattice information and cell attributes (such as current volume) (Fig. 4 a). Process I chose a trial site (marked dark), and calculate the system energy change $\triangle E$ using equation 1, the probability that this change be accepted is formulized by equation 2 (Fig. 4 b). If process I accepted the change, cell σ's information can be updated to process I easily (Fig. 4 c). Meanwhile, since process II may need the newest information of cell σ to employ Monte Carlo operations in subdomain II, this newest information should also be transfered to process II immediately. Technically, it is almost impossible to implement this immediate update in MPI-1. And the new MPI-2 [16] standard provides some support for Remote Memory Access (RMA). The implementation of one-side remote memory access (such as remote read/write) becomes easier (Fig. 4 d). But the performance issue still exist. Similar to the common update operation, there are lock/unlock

operations associated with every remote memory update. For every cells on the boundary, every successful site trial will induce an expensive remote update. As a result, there will be a lot of short messages among different processes, and the performance of the system will decrease significantly.

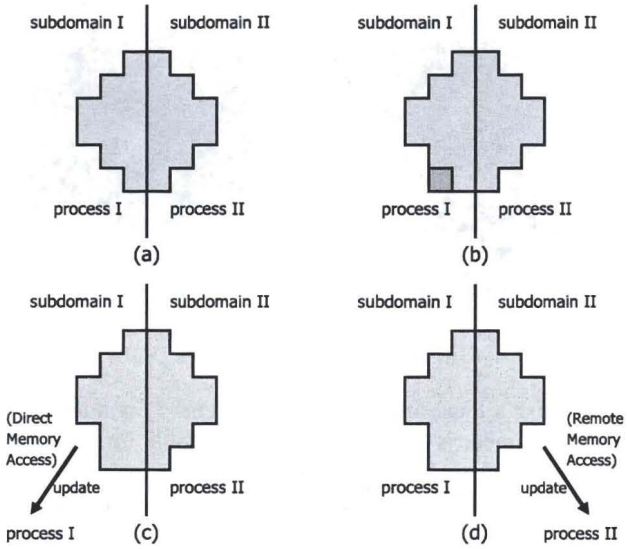


Fig. 4. Monte Carlo operation

To avoid these expensive remote memory access, we adopted OpenMP for Monte Carlo operation. OpenMP is a parallel approach for shared-memory SMPs. The computation task is shared by different processes, but the lattice data and cells' attributes are stored in shared memory which can be direct accessed by all the processes. The lock/unlock of memory accesses are handled in hardware level. In this approach, before the Monte Carlo lattice update, the master node will gather subdomain lattice information and cell information from slave nodes. Then, the master SMP node employ Monte Carlo operations parallelly using OpenMP. Upen accomplishment, master nodel will scatter subdomain data to corresponding slave nodes for next operation (Fig. 5).

2) Solving the diffusion equations: In the CPM, both tissue physiology and morphogenesis depend on diffusion of chemical morphogens in extra-cellular fluid or matrix (ECM). These natural phenomena can be represented by a general diffusion equation:

$$\frac{\partial C}{\partial t} = D\nabla^2 C + a \tag{3}$$

where C is the concentration of chemical, D is the diffusion coefficient, and a is the production or absorption rate (as local source or sink). This equation can be solved using a simple explicit method [17], which is easy to implement and is independent of the boundary condition. But to reduce the error, the duration of each Monte Carlo step must be short (in the order of seconds). This kind of explicit approach is suitable for short time simulation. To accelerate long time simulation (e.g. in the order of several months for a tumor

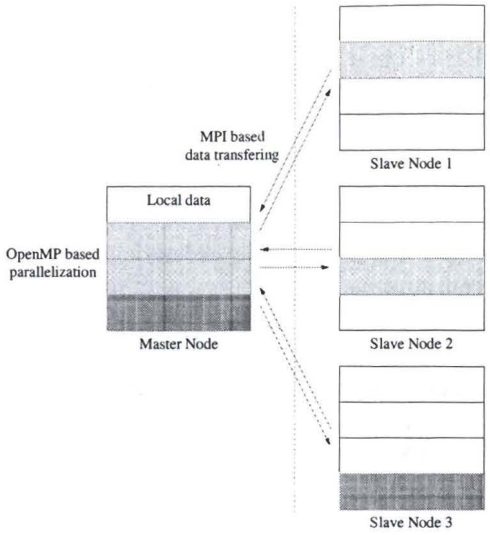


Fig. 5. OpenMP based parallelization in Monte Carlo operation

growth problem), implicit schemes are a better choice for solving the diffusion equation. In the parallel implementation of the CPM, from the content of the underlying lattice and cells' information, we can get the corresponding chemical lattices (e.g. chemical concentration, metabolic rate). Since each particular chemical lattice is monolayer, this diffusion equation can be solved parallelly using domain decomposition method togethered with either implicit scheme or explicit scheme (Fig. 6).

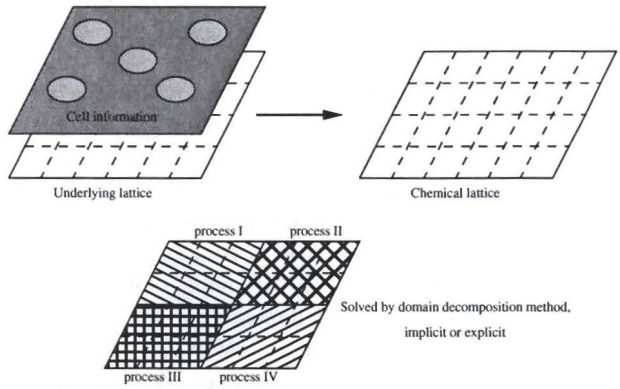


Fig. 6. Parallel solving of diffusion equation

3) Cell Division: Division is an important stage of cell evolution. In the CPM, division involves updates in both the underlying lattice layer and the top information layer. In the division operation, the volume of parental cell is exactly the same as the total volume of daughter cells (Fig. 7). In the PCPM, each cell has an attribute called off-lattice (float value) *Center of Mass (CM)*. If the CM of a particular cell σ belonged to subdomain γ (exclude the ghost zone), we define that cell σ belonged to subdomain γ. It is easy to prove that every cell belongs to one and only one subdomain. For the division operation, if cell σ belongs to subdomain γ, the division of cell σ will be computed by the process who is

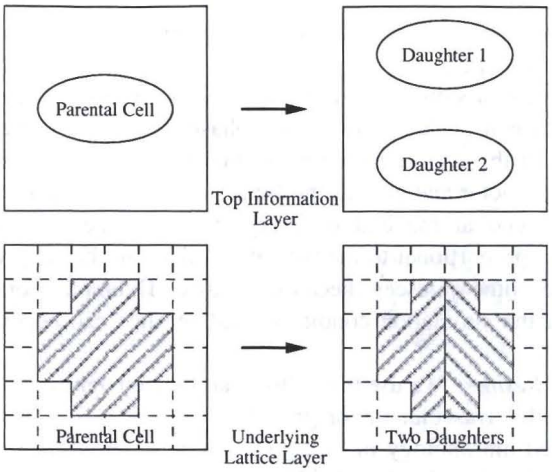


Fig. 7. Cell division in the Cellular Potts Model

responsible to subdomain γ (Fig. 8). If cell σ's information (lattice information & cell information) is required to transfer to neighbour processes, the sites' information is copied to a buffer zone in the format of quadruple structure. The first three elements of the quadruple structure are the X, Y, and Z coordinates of the site, and the fourth element is the ID number σ of the site. This strategy is state-of-the-art in avoiding inter-subdomain data inconsistency (Fig. 8). If we use the ordinary row updates rather than this quadruple update scheme, different processes will modify the same site and introduce data inconsistency (Fig. 8 d).

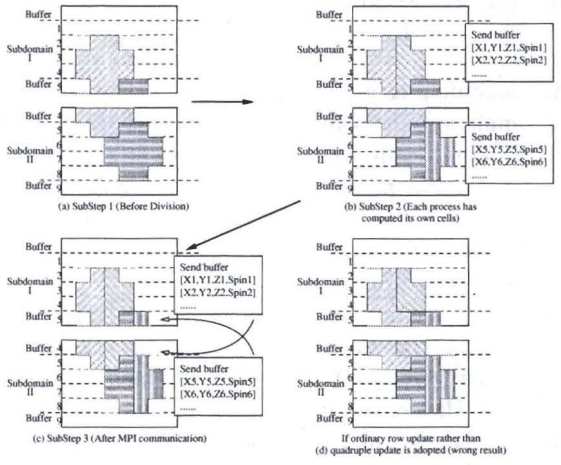


Fig. 8. Parallel implementation of division operation and quadruple update strategy

Upon division, the new daughter cells will be assigned new ID number. In the CPM, the ID number for each cell should be unique. To avoid the ID number conflict, the algorithm presented in Fig. 9 is applied when assigning new ID number.

For any ID number (S_r) assigned by rank r, $S_r \equiv r(modN)$. Let S_i be the set of all the ID numbers assigned by rank i, S_j be the set of all the ID numbers assigned by rank j. It's easy to prove that:

Step 1:Let N be the number of processes. Let r be the rank of current process in the communication group ($r = 0, 1, \ldots, N-1$). Let S_r be the ID number will be assigned to newest cell. Set $S_r = r$.
Step 2:If there is any new-born cell, the ID number for the new-born cell is set to be S_r.
Step 3:Set $S_r = S_r + N$, go to step 2.

Fig. 9. The ID assigning algorithm used in cell division operation

$$\texttt{For any } i \neq j, \quad S_i \bigcap S_j = \emptyset \tag{4}$$

By adopting this ID assigning algorithm, each process can assign its own ID numbers without any confliction and communication with other processes.

4) Cell Death: In the CPM, cell death may involve lattice update and cell information update, and unlike Monte Carlo operation, it does not need to maintain an immediate data consistent between neighbour processes. Therefore we can apply the same method applied to cell division to cell death operation. Every processor is responsible for its own cells, and inter-subdomain communications use the quadruple format (section II-D3 and Fig. 8).

5) Exception Handling: Since the width of ghost zone is closely related with performance, we cannot set the width of ghost zone (also SCW) to a large value. And because cells may have all kinds of shapes, no one can guarantee that every cell can be contained into a SCW^3 cube. Though the probability that produce these abnormal shapes (e.g., very slender) is very low, we still need to consider the exceptions. To handle these

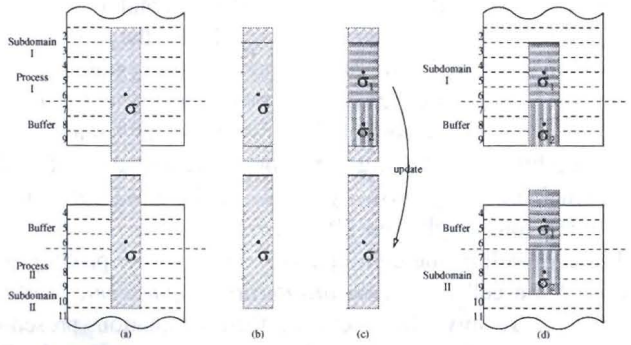


Fig. 10. The sites whose distance from Center of Mass are bigger than Safe Cell Width (SCW) are treated as medium. These outlaying sites are deleted during cell division

rare events, we assume that for a normal cell, the distances from the sites to the Center of Mass should not be more than SCW. In our implementation, there is a sphere, whose radius is SCW and center is the Center of Mass of the cell. Any sites outside this sphere are abnormal sites and treated as exception in the implementation. Fig. 10 gives an example of how these exceptions are handled. Cell σ has its Center of Mass in

subdomain I, so according to section II-D3, the division of cell σ will be calculated by process I. The SCW is equal to 3 lattice sites (Fig. 10 a). If cell σ is very long, in process I, the top sites and bottom sites are marked as abnormal sites (Fig. 10 b). The division is conducted in normal area, and cell σ is divided into two daughter cells (cell σ_1 and σ_2). Abnormal sites are deleted from lattice (converted to medium ID), and new cell information is updated to process II (Fig. 10 c). At last, the exception is handled and both processes have the correct new cell information (Fig. 10 d).

III. Application, Validation and Analysis

A. Avascular Tumor Growth

The cellular Potts model is very suitable for studying the principles of collective behavior of cells, i.e. phenomena which can not be explained at the single cell level but is arised as the result of the interactions of a population of cells. With the hybrid parallelized cellular Potts model (section II), we can simulate larger scale physical systems using much less time. Here we show as an example, we use the hybrid parallel approach to simulate the avascular tumor growth.

Typically, tumor growth is divided into three stages: *avascular tumor growth*, *angiogenesis*, and *vascular tumor growth*. In the avascular stage, the tumor develops in the absence of blood supply (hence the name). All the nutrients are supplied by diffusion from surrounding tissues. Since these tumor cells can not acquire sufficient nutrients to ensure uncontrolled multiplication, the tumor undergo a quasi-exponential growth phase followed by a saturation phase [18]. The avascular tumors typically grow into a size about 1 to 2 mm.

The tumor growth is not the result of a single cell growth, but the result of the interactions of a population of cells. To model the tumor growth naturally, we need to take into account the subcellular cell cycle, the internal protein regulatory network, the intercellular repulsion, the cell growth, and the reaction-diffusion of chemicals (e.g. oxygen, nutrients, and etc.). The intrinsic multiscale properties of tumor growth demand a multiscale model. Following the multiscale model proposed by [3], and using our hybrid parallelization of CPM, we model the the growth dynamics of the EMT6/Ro mouse mammary tumor spheroid [19], [20].

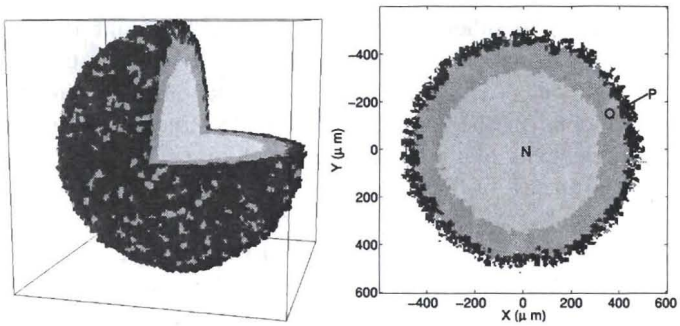
For each cell σ, the cell type $\tau(\sigma)$ refers to the proliferating status of the cell. It can be *proliferating*, *quiescent*, *necrotic*, or *medium*. To solve the reaction-diffusion equations presented in section II-D2, a full implicit scheme is adopted. The discretization of PDEs is presented below.

$$\begin{aligned} & \frac{\partial C}{\partial t} = \nabla \cdot (D \nabla C) + S \\ \Rightarrow \quad & \frac{C_{k+1} - C_k}{\Delta t} = D \nabla^2 C_{k+1} + S \\ \Rightarrow \quad & \frac{C_{k+1} - C_k}{\Delta t} = D \mathcal{L} C_{k+1} + S \\ \Rightarrow \quad & (I - \Delta t D \mathcal{L}) C_{k+1} = C_k + S \Delta t \\ \Rightarrow \quad & A C_{k+1} = b \end{aligned} \tag{5}$$

where $\mathcal{L}$ is the matrix corresponding to the discrete Laplace operator. After discretization on the structured rectangular grids, these equations are solved using the parallel BoxMG solver (Parallel Black Box Multigrid solver for 3D Symmetric Problems) [21].

We use a simplified Boolean network to model the regulation of cell cycle from G1 to S phase transition. The expressions of these proteins are controlled by the concentrations of growth factor and growth inhibitor factor. Volume checkpoints are placed at the end of every phase. If the cell has not grown proportional to the time it has lived in the cell cycle, it will become quiescent because of stress. Detailed information about the simulation conditions and settings can be found in [3].

1) Results: We use the hybrid parallelized approach to simulate the avascular tumor growth for 40 days. The simulated layered morphology of avascular tumor spheroids at the 18th day is illustrated in Fig. 11. Because of the spherical symmetry [illegible] spherical shape and 2D



(a) 3D (b) Cross-section

Fig. 11. The simulated layered morphology of avascular tumor spheroids at the 18th day.

The saturation is got on about the 30th day. According to the simulation, the saturation is achieved when there isn't any proliferating cells left. This may not be the only way of saturation. Another possible situation is when the cell division rate balances the shedding rate. We fit the simulation data to the Gompertz function, $y = a \exp(-b \exp(-cx))$, and the adjuncted R^2 is 0.9848.

The oxygen and nutrient supply as well as the accumulation of wastes and growth inhibitor factor in avascular tumor spheroids may greatly influence the cell reaction both *in vitro* and *in vivo*. Hence, the quantitative knowledge of chemical profiles may help the understanding of avascular tumor growth. A few investigations have studied the distribution of O_2, H^+ ions or nutrients in multicellular tumor spheroids. Figure 13 illustrates the O_2 profiles produced by our simulation. They fit other experimental measurement [22] and theoretical analysis [23] quite well.

B. Performance analysis

To evaluate the performance of the hybrid parallelization approach, we collected the communication traffic of the parallel avascular tumor growth simulation. The computation was run on a SMP cluster that is composed of 4 computational nodes connected with 1Gb Ethernet. Each node has quad

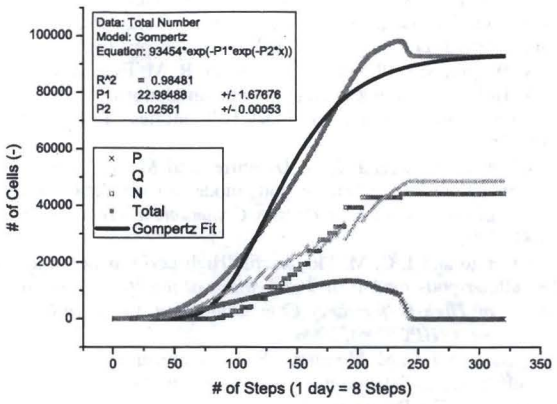


Fig. 12. Number of tumor cells as a function of simulation steps. At the end of simulation, there isn't any proliferating cells left, hence the saturation.

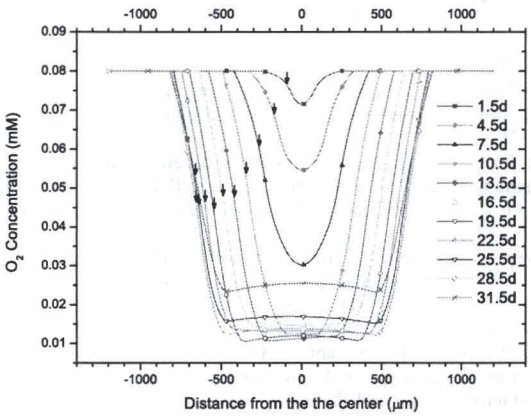


Fig. 13. Profiles of O_2 produced by our model at different time steps. d indicates days, arrows indicates the boundary between the tumor spheroids and the diffusion-depleted zones.

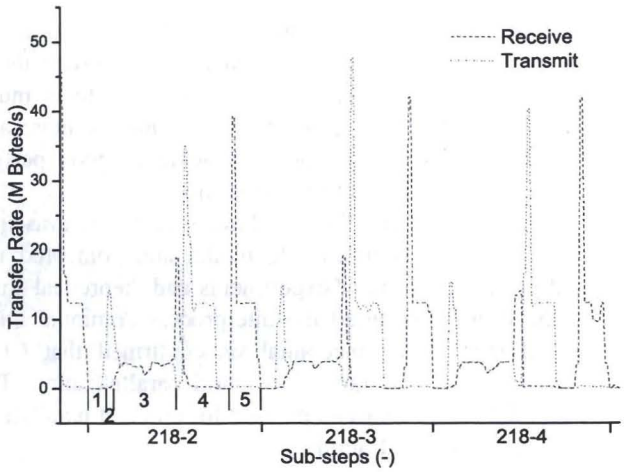


Fig. 14. The communication traffic of the master node.

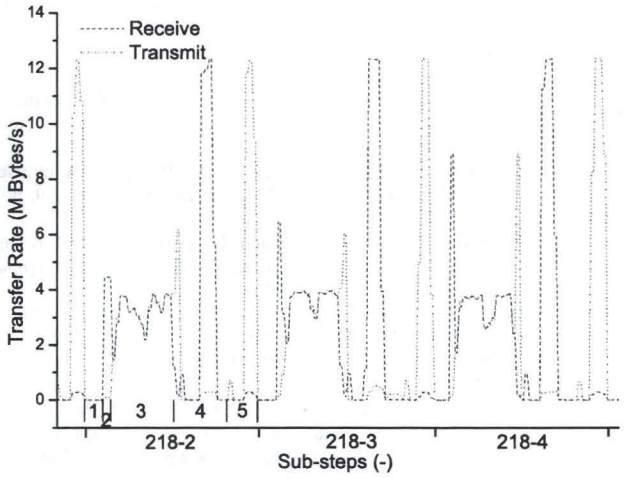


Fig. 15. The communication traffic of the slave node.

Xeon 2.4GHz processors, 8GB physical memory and RHEL operating system installed. In the application, eight processes are initialized into the four nodes. The lattice size is 400^3, and the system contains around 10^5 cells. The communication traffic of the master node and the one of the slave node are illustrated in Fig. 14 and Fig. 15 respectively. Here, data of three sub-steps in the step 218 (see Fig. 12) are presented. In the simulation, each sub-step costs about 90 seconds.

As presented in section II-D, each sub-step is composed of several operations, the time line of which is illustrated in Fig. 14 and Fig. 15. In the time interval 1, the Monte Carlo lattice update (see section II-D1) is processed. In the time interval 2, necessary data for chemical equation solving are initialized. The actual advection-diffusion equation solving (see section II-D2) is conducted in the time interval 3. The time interval 4 is for data scattering, cell reacting, and cell division. At last, the data gathering is conducted in the time interval 5. The equation solving step consumes almost 2/5 of the computation time, and the Monte Carlo lattice update is relatively fast even if it's only computed in the master node. Actually, this property holds for most CPM simulations. Because in the Monte Carlo lattice update, most operations are integer addtion, subtraction and comparison. And the equation solving is much more complex and CPU-intensive. This inherent property confirms the advantage of the hybrid approach further.

To demonstrate the capability of the hybrid parallel approach, we have also tested a "uncontrolled" case, in which cells can grow and divide without limit. In the "uncontrolled" case, the lattice size is 500^3 and the final number of cells exceed 10^6. Our system can handle them effectively, which once more proved the ability of the hybrid approach for handling large scale systems.

IV. Conclusion

In this paper, we have presented a hybrid parallelization method for cellular Potts model. In this hybrid approach, the time-consuming PDE solving, cell division, and cell reaction

operation are distributed to clusters. The Monte Carlo lattice update is parallelized on shared-memory SMP system using OpenMP. Because the Monte Carlo lattice update is much faster than the PDE solving and SMP systems are more and more common, this hybrid approach achieves good performance and high accuracy at the same time.

Based on the parallel CPM, we have studied the avascular tumor growth using a multiscale model and compared the chemical profiles with other experiments and theoretical studies. We have also measured the interprocess communication traffic, and the performance analysis confirmed that CPM applications are very suitable for hybrid parallelization. The hybrid parallel framework can be used for efficient parallelization of large scale CPM applications.

REFERENCES

[1] J. Glazier and F. Graner, "Simulation of the differential adhesion driven rearrangement of biological cells," *Physical Review E*, vol. 47, no. 3, pp. 2128 – 2154, 1993.

[2] R. Chaturvedi, C. Huang, B. Kazmierczak, T. Schneider, J. A. Izaguirre, T. Glimm, H. G. E. Hentschel, J. A. Glazier, S. A. Newman, and M. S. Alber, "On multiscale approaches to three-dimensional modelling of morphogenesis," *Journal of the Royal Society interface*, vol. 2, no. 3, pp. 237 – 253, 2005.

[3] Y. Jiang, J. Pjesivac-Grbovic, C. Cantrell, and J. P. Freyer, "A multiscale model for avascular tumor growth." *Biophysical journal*, vol. 89, no. 6, pp. 3884 – 3894, 2005.

[4] R. M. H. Merks, S. A. Newman, and J. A. Glazier, "Cell-oriented modeling of in vitro capillary development." *Lecture notes in computer science*, vol. 3305, pp. 425 – 434, 2004.

[5] A. L. Bauer, T. L. Jackson, and Y. Jiang, "A cellbased model exhibiting branching and anastomosis during tumor-induced angiogenesis," *Biophysical Journal*, vol. 92, no. 9, pp. 3105 – 3121, 2007.

[6] R. Chaturvedi, J. A. Izaguirre, C. Huang, T. Cickovski, P. Virtue, G. Thomas, G. Forgacs, M. Alber, G. Hentschel, S. A. Newman, and J. A. Glazier, "Multi-model simulations of chicken limb morphogenesis." *Lecture notes in computer science*, vol. 2659, pp. 39 – 49, 2003.

[7] A. F. M. Marée and P. Hogeweg, "How amoeboids self-organize into a fruiting body: Multicellular coordination in dictyostelium discoideum," *Proc. Natl. Acad. Sci. USA*, vol. 98, no. 7, pp. 3879–3883, 2001.

[8] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller, "Equation of State Calculations by Fast Computing Machines," *Journal of Chemical Physics*, vol. 21, pp. 1087–1092, 1953.

[9] G. T. BARKEMA and T. MACFARLAND, "Parallel simulation of the ising model," *Physical review. E, Statistical physics, plasmas, fluids, and related interdisciplinary topics*, vol. 50, no. 2, pp. 1623 – 1628, 1994.

[10] S. A. Wright, S. J. Plimpton, T. P. Swiler, R. M. Fye, M. F. Young, and E. A. Holm, "Potts-model grain growth simulations: Parallel algorithms and applications," Sandia National Laboratories, Tech. Rep., August 1997.

[11] N. Chen, J. A. Glazier, J. A. Izaguirre, and M. S. Alber, "A parallel implementation of the cellular potts model for simulation of cell-based morphogenesis," *Computer Physics Communications*, vol. 176, pp. 670 – 681, 2007.

[12] F. P. Cerato and J. C. M. Mombach, "High performance simulations of the cellular potts model," in *Proceedings of the 20th International Symposium on High-Performance Computing in an Advanced Collaborative Environment (HPCS'06)*, 2006.

[13] Éder Gusatto, J. C. M. Mombach, F. P. Cercato, and G. G. H. Cavalheiro, "An efficient parallel algorithm to evolve simulations of the cellular potts model," *Parallel Processing Letters*, vol. 15, pp. 199–208, 2005.

[14] W. Gropp, E. Lusk, and A. Skjellum, *Using MPI: Portable Parallel Programming with the Message Passing Interface,* 2nd edition. Cambridge, MA: MIT Press, 1999.

[15] R. Chandra, L. Dagum, D. Kohr, D. Maydan, J. McDonald, and R. Menon, *Parallel programming in OpenMP.* San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2001.

[16] W. Gropp, E. Lusk, and R. Thakur, *Using MPI-2: Advanced Features of the Message-Passing Interface.* Cambridge, MA: MIT Press, 1999.

[17] D. Dan, C. Mueller, K. Chen, and J. A. Glazier, "Solving the advection-diffusion equations in biological contexts using the cellular potts model," *Phys. Rev. E*, vol. 72, no. 041909, October 2005.

[18] J. Folkman and M. Hochberg, "Self-regulation of growth in three dimensions," *Journal of Experimental Medicine*, vol. 138, no. 4, pp. 745–753, 1973.

[19] J. P. Freyer and R. M. Sutherland, "Proliferative and clonogenic heterogeneity of cells from EMT6/Ro multicellular spheroids induced by the glucose and oxygen supply," *Cancer Research*, vol. 46, no. 7, pp. 3513–3520, 1986.

[20] J. Freyer, "Regulation of growth saturation and development of necrosis in EMT6/Ro multicellular spheroids by the glucose and oxygen supply," *Cancer Research*, vol. 46, no. 7, pp. 3504–3512, 1986.

[21] J. Dendy Jr, "Two multigrid methods for three-dimensional problems with discontinuous and anisotropic coefficients," *SIAM Journal on Scientific Computing*, vol. 8, no. 5, pp. 673–685, 2006.

[22] W. Mueller-Klieser and R. Sutherland, "Influence of convection in the growth medium on oxygen tensions in multicellular tumor spheroids." *Cancer Res*, vol. 42, no. 1, pp. 237–42, 1982.

[23] K. Groebe and W. Mueller-Klieser, "Distributions of oxygen, nutrient, and metabolic waste concentrations in multicellular spheroids and their dependence on spheroid parameters," *European biophysics journal*, vol. 19, no. 4, pp. 169 – 181, 1991.