Developing a Scalable Cellular Automaton Model of 3D Tumor Growth

Cyrus Tanade[1 [0000−0002−2395−6908]], Sarah Putney[1], and Amanda Randles[1 [0000−0001−6318−3885]]

Duke University, Durham NC 27710, USA
{cyrus.tanade, sarah.putney, amanda.randles}@duke.edu

Abstract. Parallel three-dimensional (3D) cellular automaton models of tumor growth can efficiently model tumor morphology over many length and time scales. Here, we extended an existing two-dimensional (2D) model of tumor growth to study how tumor morphology could change over time and verified the 3D model with the initial 2D model on a per-slice level. However, increasing the dimensionality of the model imposes constraints on memory and time-to-solution that could quickly become intractable when simulating long temporal durations. Parallelizing such models would enable larger tumors to be investigated and also pave the way for coupling with treatment models. We parallelized the 3D growth model using N-body and lattice halo exchange schemes and further optimized the implementation to adaptively exchange information based on the state of cell expansion. We demonstrated a factor of $20x$ speedup compared to the serial model when running on 340 cores of Stampede2's Knight's Landing compute nodes. This proof-of-concept study highlighted that parallel 3D models could enable the exploration of large problem and parameter spaces at tractable run times.

Keywords: cellular automaton · parallel computing · tumor growth.

1 Introduction

Three-dimensional (3D) models of tumor growth that leverage parallel computing can provide a means to explore 3D multiscale tumor morphology efficiently. Understanding tumor growth dynamics is fundamental in cancer biology, and parallelized 3D models can efficiently capture large-scale dynamics. Mathematical models of cancer biology are increasingly being used to understand tumorigenesis, metastasis, and responses to treatment [1].

Cancer is fundamentally defined by the uninhibited growth of cells. Models that capture cell-cell interactions that span from individual cells to emergent tumors are needed to better understand different features that influence growth dynamics. Cellular automaton models represent cells as dead or alive and interact within a fixed local neighborhood. Agent-based models (ABMs) can have multiple states and incorporate complex interaction networks beyond local neighbors. Game of Life (GoL) constructs offer the capability to use simple rules that are

ICCS Camera Ready Version 2022
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-08751-6_1

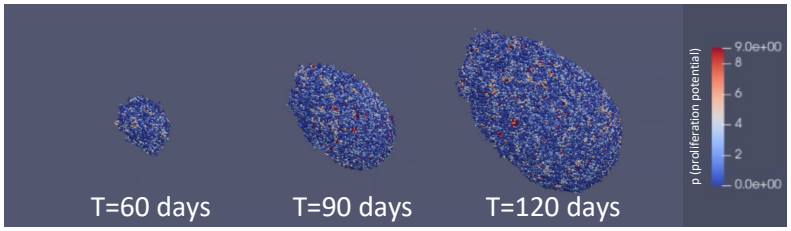

Fig. 1. 3D Tumor Growth Over Time. Tumor growth over 60, 90, and 120 days displaying proliferation potential (0-low, 9-high).

fundamentally rooted in single-cell kinetics to observe emergent dynamics that could span from a few initial seed cells to millions of cells [2, 3]. Dissimilar to multi-agent systems - where agents are used to solve a specific problem, GoL and ABM models are used to study emergent behavior. However, implementations of GoL models have conventionally been restricted to 2D due to computational complexity and associated burden [4, 5, 6].

The development of tumors resembles Darwinian evolution where cancer cells would need to compete for resources and space. GoL models of cancer cell growth are relevant in the pre-angiogenic phase when tumor growth is driven by cell-cell interactions in small neighborhoods of cells. These GoL models have been shown to model tumor growth well despite simple rules and small parameter spaces [4, 7]. However, as these models are expanded from two-dimensional (2D) to three-dimensional (3D) models, the computational space grows exponentially and could become intractable to simulate [6]. Exploring certain applications, such as studying how 3D morphology changes in response to treatment, is only amenable for 3D models and inherently requires more memory than 2D models such that simulations could become limited by the size or duration of tumor growth. Here, we extended a 2D cellular automaton model from Poleszczuk-Enderling [6] to recapitulate 2D tumor dynamics on a slice-level and output 3D growth dynamics for tumor morphology studies. Furthermore, we parallelized the model to realize larger problem sizes.

There are inherent limitations of 2D models. *In vitro* and *in silico* 2D models could reliably mimic *in vivo* tumor dynamics of a range of cancers, and as a result, have been instrumental for understanding cancer cell physiology. However, 2D monolayer representations of tumor growth could fail to recapitulate *in vivo* proliferation, morphology, and cell-cell and cell-matrix interactions. 2D cell culture models lack complex 3D architecture and extracellular-matrix interactions that are pervasive *in vivo*. While there are sophisticated techniques for efficiently capturing the 2D dynamics of tumor growth and proliferation, we found a lack of computationally efficient, scalable methods to capture the 3D morphology. Our study aims to bridge this gap by developing a parallel 3D model of tumor growth. In this study, we developed a proof-of-concept multiscale 3D model that can simulate from a few seed cells to millions of interacting cells (**Figure** 1). Such

ICCS Camera Ready Version 2022
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-08751-6_1

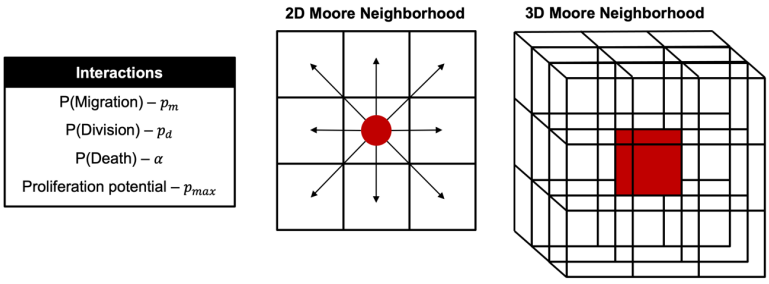

Fig. 2. 3D Model Design. The Poleszczuk-Enderling 2D model used a 2D Moore Neighborhood, where the center cell could interact with it's immediate 8 neighbors. Extending the model to 3D resulted in a 3D Moore Neighborhood with 26 neighbors. Cells have a finite proliferation potential and could interact with the neighborhood via migration, division, or death.

a model could serve as a virtual 3D cell culture and enable low-cost investigation of cell pathophysiology and drug discovery.

In this model, cell movement was based on a Monte Carlo implementation. An ensemble of simulations must be completed to buffer randomness in the model, which further highlights the overarching need to reduce time-to-simulation for one individual simulation instance. The Poleszczuk-Enderling 2D tumor growth model took 3-5 hours of runtime for 4 months of tumor growth, and running multiple instances to get averaged results could scale quickly.

We demonstrated the ability to capture 3D dynamics and propose techniques for efficient parallelization. As such, we made the following contributions: (1) development of a 3D cellular automaton model, (2) parallelization of a 3D serial model using MPI and optimizations to minimize memory transfer between processes, and (3) performance evaluation of the proof-of-concept parallelization scheme up to 340 tasks (or 5 nodes) on Stampede2 Knight's Landing (KNL) compute nodes. Our results demonstrated that the model could be most useful for simulating dense tumors with a large cell count and long durations.

2 Related Work

3D cellular models can serve as virtual laboratories with fully tunable conditions that enable the investigation of emergent tumor behavior. 3D cellular automaton models have been used to study tumor microenvironments and treatment paradigms, but are usually tuned to one cancer type of interest [8, 9, 1]. ABMs have been used to explicitly model adhesive, locomotive, drag, and repulsive forces between cells and have been applied to model cellular responses to hypoxia in breast cancer. Such models have been shown to scale to millions of cells [10]. There are also works parallelizing the Poleszczuk-Enderling model, but are limited to thread-level parallelism on local machines [11]. The cellular automaton model in this work is designed to be agile by relying on minimal

ICCS Camera Ready Version 2022
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-08751-6_1

input parameters and thereby easily be tuned to different types of cancers [6] to model single-cell kinetics from a few initial cells to millions of cells. Even so, actions of the individual cells rely on random sampling to drive their interactions. The inherent stochasticity requires simulation of a large ensemble of potential interactions to adequately capture the macroscopic behavior of the tumor. The computational burden of an individual instance is exacerbated by the need to complete many simulations to account for the underlying stochasticity. Moreover, current models are typically limited by the size of the tumor that can be simulated. We addressed these challenges through a multi-level parallelization scheme targeting 3D tumor models.

3 Methods

3.1 Extension of 2D Cellular Automaton Model to 3D

The Poleszczuk-Enderling model introduced a 2D representation of tumor growth relying on a cellular automaton representation of cancer cells [6]. Tumor growth was captured via cellular interactions, where each cell was modeled as an individual agent. The interactions characterizing cell growth included migration to other discrete lattice points, proliferation via mitotic cell division, and cells could finally die or become quiescent. As proliferation and migration require moving to a different lattice point, communication was needed within a cell's neighborhood to determine if there were empty spaces for interaction. The Poleszczuk-Enderling 2D model used a 2D Moore Neighborhood, where cells could interact with 8 of it's immediate neighbors (**Figure** 2). The model included cancer stem cells and non-stem cancer cells. Stem cells were assumed to have infinite proliferation potential, whereas the non-stem cells had a maximum proliferation potential (p_{max}). Cells interacted in the lattice via discrete lattice-based rules governed by probabilities of migration (p_m), proliferation (p_d), and death (α). Each of these traits was kept as trait vectors for each cell such that there were N-body cells. On the other hand, the number of free spots in each neighborhood was stored on the lattice. N-body and lattice components of the simulation were eventually parallelized separately.

Expansion from 2D to 3D consisted of transitioning the cell lattice, which tracks the number of empty nearest neighbors for a given grid location, from a 2D array to a flattened 3D vector. Helper functions for cell death, proliferation, and migration were adjusted to account for a 3D Moore's Neighborhood of 26 neighbors rather than the initial 2D Moore's neighborhood. We retained input parameter values from the 2D model, but increased the cell division probability (p_d) by 30 %. Though this was not a necessary adjustment, it provided the benefit of creating tumors comparable in cross-sectional density to the Poleszczuk-Enderling 2D code and allowed comparison of 2D slices across the center of mass with 2D model outputs. Additionally, this process demonstrated the ease of model tuning.

The time loop was iterated over a user-specified number of time steps (where each step was a simulation of one hour). Within each time step, every cell in the

ICCS Camera Ready Version 2022
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-08751-6_1

population was iterated over. Random number generation was used to first check if the cell should spontaneously die and be removed from the population. If the cell did not die, a new random number was generated to determine if the cell should proliferate. Only stem cells or cells that have not reached proliferative exhaustion could divide. If the cell did not proliferate, the random number was checked to determine if the cell should migrate. If the cell did not die, proliferate, or migrate, then nothing would happen and the cell would be added back into the population vector for the next time loop. For each cell, the lattice containing the number of empty nearest neighbors at each grid point was updated based on the cell's actions. After iterating over all the cells, the vector holding the population was refreshed to remove any dead cells and add any new ones.

3.2 N-body and Lattice Parallelization Schemes

Parallelization of the main simulation loop took two forms - an N-body scheme for individual cells and halo exchanges for the underlying lattice. The cell lattice domain was divided into approximately even blocks along the $x\ axis$, where each rank was responsible for a local cell lattice of size $lx_{local} * ly * lz$ (where lx_{local} was the length of the x domain local to the rank, ly was the length of the y domain, and lz was the length of the z domain). We were limited to communicating 2D packets of 3000x3000 points because of inherent limits to the size of 2D C++ vectors. A potential future direction would be to optimize these messages further.

Cell movement and cell lattices were parallelized separately. Cells were allowed to move freely into empty neighbors until reaching the edge of their domain, where we implemented a one layer overlap between each rank. This overlap was used as a part of domain decomposition for parallelization and did not influence the mechanics of tumor growth. Cells migrating outwards at the edge of a task's boundary would be transferred to neighboring tasks. This particular overlap allowed cells to move freely into empty spaces, only triggering communication between ranks when a cell entered border regions where a full Moore's neighborhood could not be realized. MPI communication of the cells between ranks occurred in two parts: all the integer properties associated with the moved cells were sent, and then the characters associated with the cells were sent. Communication was implemented using non-blocking sends and receives with the receives posted at the start of each time step and the sends posted at the end of each time step.

The cell lattice was represented as a flattened vector of integers denoting the number of free spaces a cell at a given index had around it. When a cell died, proliferated, or migrated, the lattice values of the surrounding Moore's neighborhood must be updated accordingly. Instead of one layer of overlap between ranks (like that used in the N-body communication), we used an additional layer of communication on each rank that would track a cell's movement into and out of the cell transfer zone. At the end of each time step, before the border cells from neighboring ranks were transferred, the lattice values for these two layers were communicated, compared, and reconciled such that when the data from the

ICCS Camera Ready Version 2022
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-08751-6_1

N-body-based communication were added into the rank population, they could access values from an appropriately updated lattice.

Based on our division of ranks along the x *axis*, we placed the initial cell in the simulation in the middle of the median rank. Verification of the parallelized code with the serial code was performed using both odd and even number of ranks.

3.3 Adaptive Communication Scheme to Reduce Overhead

The naïve 3D parallelized model communicated cell and lattice data between each rank in a point-to-point complete manner. Even ranks that contained no cells participated in this communication, which unnecessarily increased memory transfer and associated communication costs. To optimize the communication scheme, before any lattice/cell communication, we gathered across every rank an array of Boolean values indicating the ranks that contained cells and then used this array to determine which ranks needed to communicate. Each rank that had cells would send to both its nearest neighbors, and each rank would receive from any nearest neighbor that contained cells. This optimization meant that ranks still participated in point-to-point communication, but communication of the entire size of the send buffer (up to 2^{22} elements) only occurred when there were actually cells on the rank that necessitated this communication. As a result, the communication expanded adaptively as cells propagated across ranks over time.

3.4 3D Serial and Parallel Verification Protocols

We verified our 3D model by comparing per-slice cell population sizes with the 2D simulation at 60 days of simulation. The morphology was compared using the dice similarity coefficient (DSC), which measures the spatial overlap of the tumor. We verified our parallel code by comparing it to the serial code. To do so, we needed to consider the inherent stochasticity of our simulation model which used multiple randomly generated numbers per cell in the population per time step. While the simulation was capable of being seeded to generate the same tumor population for runs of the same world size, seeding each rank individually would not yield the same results. Consequently, we verified the code by defining 4 variables of interest - cell population size, the proportion of stem cells, the distribution of cell proliferation potential across the population, and the distribution of the number of empty nearest neighbors across the population. We used world sizes of 1, 16, 17, 54, and 55 ranks when completing verification runs and performed 5 different unseeded runs for each case. We calculated the mean and standard error for the 4 variables of interest on the 80th day of simulation.

3.5 Performance Evaluation of the Parallelized 3D Model

The parallelized model was evaluated for performance through strong scaling, throughput, and efficiency of using parallel resources. For all performance evaluation runs, we ran simulations of up to 120 days of tumor growth and up to

ICCS Camera Ready Version 2022
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-08751-6_1

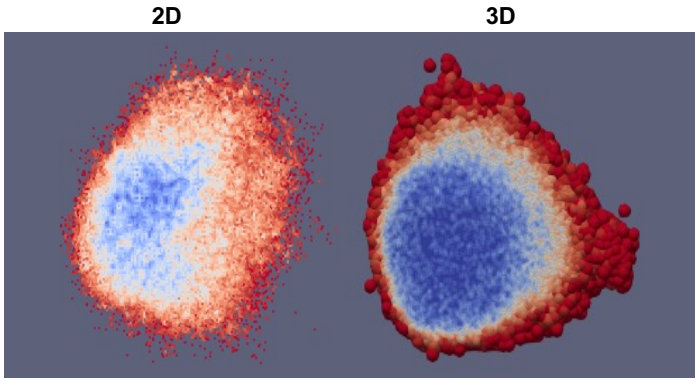

Fig. 3. 3D Model Verification. Per-slice population counts through the center of mass at 60 days resulted in a mean DSC score of 0.94.

340 cores of Intel Xeon Phi cores (clock rate of 1.40 GHz) or 5 nodes on Stampede2 KNL compute nodes with 100 Gb/s Intel Omni-Path network with fat tree topology interconnect. All code were compiled using the `-O3` optimization flag. These resources were accessed at TACC via an XSEDE allocation [12]. We simulated up to 120 days because this was a similar time scale used in the Poleszczuk-Enderling 2D model [6].

Strong scaling curves were generated by computing the speedup with an increasing number of ranks for 120 day simulations. However, we found that the serial 3D code was inherently faster for the first 2-3 months of simulation time through some initial testing. This was likely due to the low number of cells in the initial stages of growth and that parallelization only became necessary once the tumor size reached a threshold. To test this observation, we measured strong scaling when neglecting the first 90 days of simulation time to illustrate that the parallel implementation needed to ramp up before eventually outperforming the serial 3D model. Ideal scaling was taken as the number of processors containing cells, averaged over the strong scaling runs due to stochasticity in the model.

We also investigated the throughput of the model by computing the cellular operations per second ($CLOPs$) for different core counts (up to 340 ranks) at different points in time (i.e., 60, 90, and 120 days). Lastly, the efficiency - the number of ranks containing cells relative to the total number of ranks used in the simulation - was measured across all time points and averaged across all the different runs with different number of ranks. This performance measure would indicate the rate of uptake of ranks and the increase in efficiency over time. As the simulations were inherently stochastic, we ran 5 unseeded simulations per data point to compute for means and standard errors.

ICCS Camera Ready Version 2022
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-08751-6_1

4 Results

4.1 3D Model Preserves Per-Slice Tumor Morphology

The 3D model agreed with the 2D model on a per-slice basis through the center of mass of the tumor (**Figure** 3). The comparison was made at 60 days of simulation. The DSC was 0.94, which demonstrated that the morphology was successfully preserved after increasing the dimensionality of the model and changing input parameters.

4.2 3D Parallel Model Matches 3D Serial Model

We first verified the final proportion of stem cells over varying world sizes (**Figure** 4A). The ideal plot would be a horizontal line, indicating that the mean proportion did not change with the number of ranks used. Though the experimental values deviated slightly from a perfect horizontal line, the stem cell proportion values for each world size were within each points' standard error range. There was a similarly horizontal trend for the final cell population size.

Next, we verified the distribution of empty neighbors and cell proliferation potential. **Figure** 4B demonstrates that the shapes of the distributions for each world size were all similar. The distribution of cell proliferation potential also resulted in similar trends. Therefore, we have verified the parallelized 3D model using the proportion of stem cells, final cell population size, distribution of empty neighbors, and the distribution of cell proliferation potential.

4.3 Strong Scaling Indicates When to Launch the Parallel Model

The parallelized 3D model had relatively modest strong scaling results (**Figure** 5). At 120 days of simulation time, there was a factor of 10 speedup compared to the serial model which was roughly equivalent to a parallel efficiency of 20 %. Parallel efficiency was taken to be relative to the number of ranks used over the total number of ranks allocated.

Such performance was because the serial model was inherently faster at the beginning of the simulation due to communication overhead. In the early steps of the simulation, the tumor was in the initiation phase going from only a few cells to many and favored the serial code because there were no communication overheads. As the tumor grew and expanded in domain to neighboring ranks, the parallel code would eventually become more efficient than the serial model. The workload demand didn't necessitate use of multiple cores at small tumor sizes. We quantified the speedup when the simulation had surpassed this initiation threshold to test these observations. We assumed *a priori* that 90 days would exceed the initial growth stage. Using this offset, we achieved a factor of $20x$ speedup with a parallel efficiency of 40 % at 120 days. This result demonstrated a trade-off between the serial and parallel code, and that there would be an optimum point of switching between the two models. The strong scaling curves

ICCS Camera Ready Version 2022
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-08751-6_1

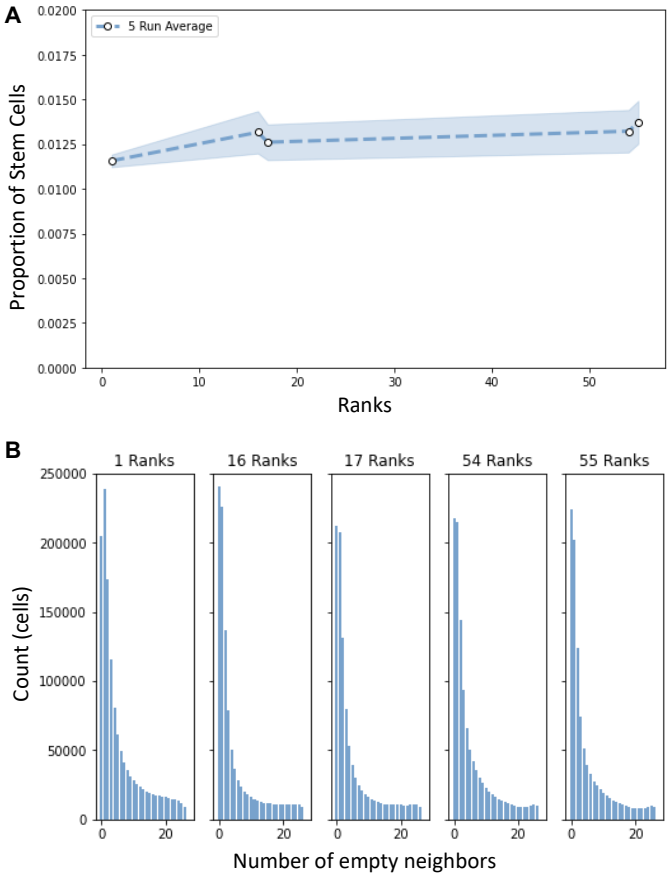

Fig. 4. Verification of Parallelized Model. 80-day simulations of varying world sizes, from 1 to 55 ranks, had consistent (A) stem cell proportion of population and (B) distributions of empty nearest neighbors. Results were averaged over 5 unseeded runs: (A) *average = dotted line, standard error = background area*; (B) *only averages were presented as standard error bounds were too small to visualize.*

had not reached the point of diminishing returns, indicating that running for longer durations could result in better performance.

The serial model was shown to be more efficient for the first few months of simulation until reaching larger cell counts and longer simulation durations, where the parallel model would eventually become more efficient. To quantify a global, world size-invariant cut-off at which this transition occurs, we computed speedup (relative to the serial model) over all time points across all simulations with different world sizes (**Figure** 6). The results indicated that the global cut-off occurred at 68 days, which suggested that the parallel code would provide gains for simulations with durations exceeding this cut-off.

ICCS Camera Ready Version 2022
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-08751-6_1

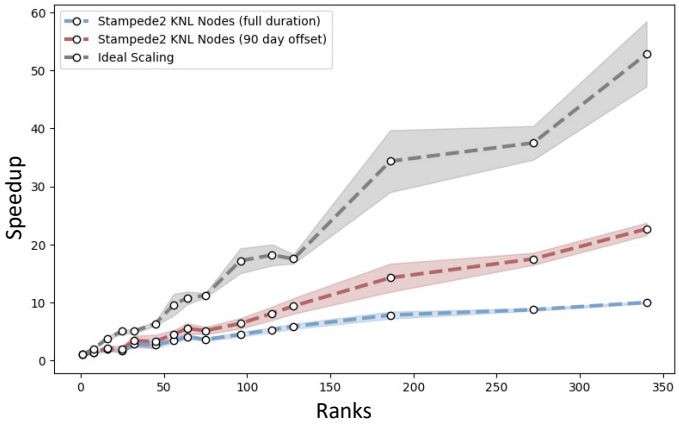

Fig. 5. Strong Scaling. Neglecting initial parts of the simulated duration resulted in improved scaling. Ideal scaling was taken as the number of ranks containing cells, averaged over 5 runs. *Average = dotted line, standard error = background area.*

4.4 Throughput and Efficiency Increases Over Time

We defined $CLOPs$ as a measure of the parallel model's throughput. The $CLOPs$ increased as the number of ranks and the number of simulation days was increased (**Figure** 7). The parallel code revealed that the model was capable of over 2 million $CLOPs$ at 120 days of simulation. From the serial model, the results at 60 days demonstrated some loss in performance, which might indicate that the serial code was more efficient than the parallel code at this point of the simulation. On the other hand, the 90 day simulation exceeded the serial run, but similarly also plateaued in $CLOPs$ when the number of ranks increased. The 120 day simulation indicated an increasing trend even at the highest number of ranks, which suggests that the parallel model is more amenable for larger problem sizes over long simulation periods.

We also investigated the efficiency of the number of ranks used relative to all the ranks assigned over time (**Figure** 8). In spite of stochasticity at the 120 day time point, the change in percentage rank utilization (or efficiency) was consistent across simulations of different world sizes. At best, the efficiency was just under 20 % after 120 days of simulation. The rise in efficiency has not reached the point of diminishing returns and serves as a lower limit of performance.

5 Discussion and Conclusion

3D models have the potential to inform cancer research, however, there is a hard limit to both the domain sizes and runtimes that serial implementations can capture. This proof-of-concept study extended a 2D model of cellular automaton of dense tumor growth to 3D and parallelized the model in one direction.

ICCS Camera Ready Version 2022
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-08751-6_1

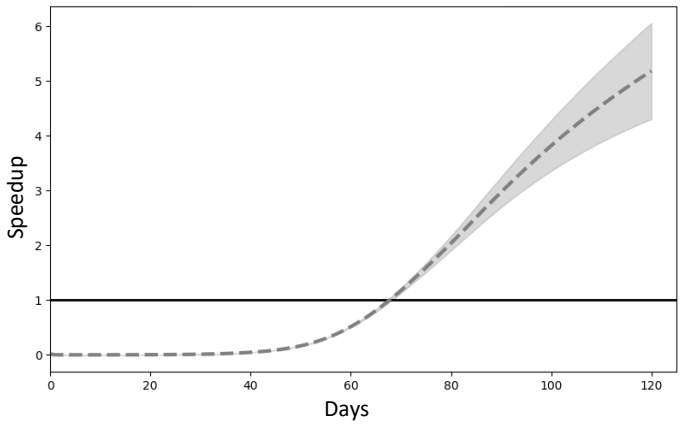

Fig. 6. Cut-off Point for Parallel Model. The serial code was initially faster than the parallel code, where speedup relative to the serial implementation was close to 0 until 50 days. The benefits of the parallel implementation became evident starting at 68 days when speedup crossed unity. The results were aggregated over multiple runs ranging from 8-340 tasks to obtain a global cut-off. *Average = dotted line, standard error = background area, baseline speedup relative to serial code = horizontal solid line.*

The results indicated tangible speedup gains that enable larger and longer simulations. The 3D parallel model verified well with the serial 3D model regarding cell population size, proportion of stem cells, proliferation potential distribution, and neighboring spaces distribution. The parallelized model was verified with an even and odd number of ranks and demonstrated that the variability between the two models was within the margin of error due to stochasticity. Moreover, the per-slice cell population size and morphology between the 2D and 3D models were comparable.

The parallelized model was evaluated for performance up to 340 tasks on Stampede2 and could result in a $20x$ speedup. It was evident that the 3D model needed to ramp up in terms of rank utilization until it would eventually overtake the serial model. We tested this observation by comparing speedup gains with and without a time delay. We found that the global cut-off point at which the parallel model would provide speedup over the serial model was at 68 days of simulation time. Parallelization increased the lattice size that could fit in memory, from 2^{31} to around 2^{34} elements - a factor of 2^7 increase. Although scaling results have not generally plateaued, we chose to simulate comparable time scales with the Poleszczuk-Enderling model [6] and show a lower limit of strong scaling, throughput, and efficiency results.

There are future paths for additional parallelization and model development. Although we enabled larger domains to be simulated, we only parallelized along the x dimension. Parallelizing over additional axes would decrease the grid sizes allocated per rank and the size of buffers to be communicated between ranks.

ICCS Camera Ready Version 2022
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-08751-6_1

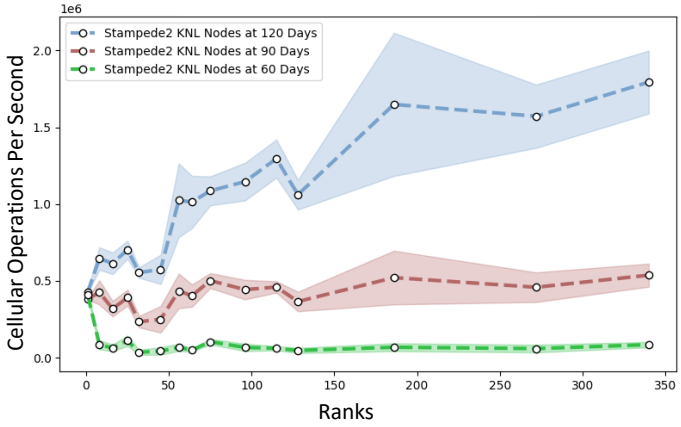

Fig. 7. Scaling cellular operations per second (CLOPs). We defined CLOPs as a measure of throughput. The CLOPs increased with longer simulations and more processors, which may indicate that the model was better suited for larger problem sizes. *Average = dotted line, standard error = background area.*

Though more communication would be required per rank, the size of buffers would be smaller and would provide speedup over the current implementation. In terms of biology, allowing cells to occupy multiple lattice points to model cellular volume expansion could provide a more detailed representation of tumor growth. Speedup could improve with such an implementation as cells may expand over the domain quicker to achieve better rank utilization. This work does not intend to provide an optimal parallel tumor growth model, but offers an evaluation of strategies that warrant further investigation.

The results indicated that the serial model was more efficient for the first 68 days of simulation, but that the model became exponentially slower from that cut-off point onward. This was likely the point at which the cells were starting to expand to a sufficient proportion of ranks that resulted in enough speedup to overcome inherent MPI communication overheads. Therefore, some speedup could be gained by restricting simulations to use the serial code until the cut-off point to avoid unnecessary communication. Checkpointing could be implemented where the serial code could stop at the cut-off and use its outputs to launch parallel tasks.

The performance evaluation also revealed that less than 20 % of ranks were actually engaging in the simulation for up to 120 days. This was a clear bottleneck, and ramping up the uptake of ranks faster would likely improve results. Although highly stochastic, the results demonstrated that even across simulations with a different number of tasks, the uptake in ranks were largely consistent. For this particular problem, it could be useful to change the thickness (or proportion of the lattice domain) handled by each rank. Instead of having relatively uniform allocation, it would be useful to allocate as many ranks as possible to the middle

ICCS Camera Ready Version 2022
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-08751-6_1

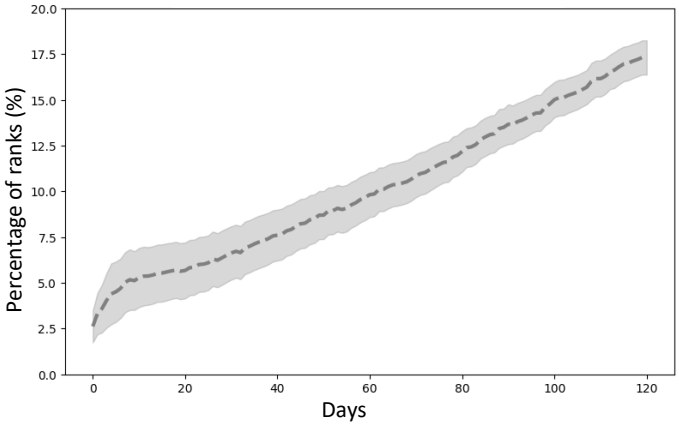

Fig. 8. Efficiency in Rank Utilization. There was a steady increase in efficiency over time. Efficiency was the number of ranks that contained cells relative to the total number of ranks allocated. *Average = dotted line, standard error = background area.*

20 % of the lattice (i.e., allocate as small of a lattice as possible) and allocate as few ranks as possible (i.e., allocate as much of the lattice as possible) to the edge ranks where cells were unlikely to reach.

Ultimately, the 3D parallelized code verified well with the 3D serial code. The multi-level parallelization scheme of combining N-Body and halo paradigms alongside adaptive communication enabled efficient simulation of 3D tumor growth. This pilot study exemplified that even with parallelization across just one direction, there were clear gains that could enable larger studies and questions to be explored. We have additionally outlined some directions that could provide more significant speedups and allow for even longer simulations. This work could lay the groundwork for future studies of cellular automaton tumor growth models and parallelization methods of computational N-body and lattice-based models.

Acknowledgements We thank Daniel Puleri, Simbarashe Chidyagwai, Sayan Roychowdhury, and Raveena Kothare for fruitful discussions. This work used an Extreme Science and Engineering Discovery Environment (XSEDE) allocation, which is supported by National Science Foundation grant number ACI-1548562. This work used Stampede2 at TACC through allocation TG-MDE210001. The research of Cyrus Tanade was supported by the National Science Foundation Graduate Research Fellowship Program under Grant No. NSF GRFP DGE 1644868. The work of Amanda Randles was supported by the National Institutes of Health under award number U01CA253511. The content is solely the responsibility of the authors and does not necessarily represent the official views of the NIH or the NSF.

ICCS Camera Ready Version 2022
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-08751-6_1

References

1. Randles, A., Wirsching, H.-G., Dean, J.A., Cheng, Y.-K., Emerson, S., Pattwell, S.S., Holland, E.C., and Michor, F.: Computational modelling of perivascular-niche dynamics for the optimization of treatment schedules for glioblastoma. Nature Biomedical Engineering 5(4), 346–359 (2021)
2. Tang, J., Fernandez-Garcia, I., Vijayakumar, S., Martinez-Ruis, H., Illa-Bochaca, I., Nguyen, D.H., Mao, J.-H., Costes, S.V., and Barcellos-Hoff, M.H.: Irradiation of juvenile, but not adult, mammary gland increases stem cell self-renewal and estrogen receptor negative tumors. Stem Cells (Dayton, Ohio) 32(3), 649–661 (2014)
3. Gao, X., McDonald, J.T., Hlatky, L., and Enderling, H.: Acute and fractionated irradiation differentially modulate glioma stem cell division kinetics. Cancer Research 73(5), 1481–1490 (2013)
4. Piotrowska, M.J., and Angus, S.D.: A quantitative cellular automaton model of in vitro multicellular spheroid tumour growth. Journal of Theoretical Biology 258(2), 165–178 (2009)
5. Jiao, Y., and Torquato, S.: Emergent Behaviors from a Cellular Automaton Model for Invasive Tumor Growth in Heterogeneous Microenvironments. PLOS Computational Biology 7(12), e1002314 (2011)
6. Poleszczuk, J., and Enderling, H.: A High-Performance Cellular Automaton Model of Tumor Growth with Dynamically Growing Domains. Applied Mathematics 5(1), 144–152 (2014)
7. Morton, C.I., Hlatky, L., Hahnfeldt, P., and Enderling, H.: Non-stem cancer cell kinetics modulate solid tumor progression. Theoretical Biology and Medical Modelling 8(1), 48 (2011)
8. Norton, K.-A., Jin, K., and Popel, A.S.: Modeling triple-negative breast cancer heterogeneity: Effects of stromal macrophages, fibroblasts and tumor vasculature. Journal of Theoretical Biology 452, 56–68 (2018)
9. Alfonso, J.C.L., Jagiella, N., Núñez, L., Herrero, M.A., and Drasdo, D.: Estimating Dose Painting Effects in Radiotherapy: A Mathematical Model. PLOS ONE 9(2), e89380 (2014)
10. Ghaffarizadeh, A., Heiland, R., Friedman, S.H., Mumenthaler, S.M., and Macklin, P.: PhysiCell: An open source physics-based cell simulator for 3-D multicellular systems. PLOS Computational Biology 14(2), e1005991 (2018)
11. Salguero, A.G., Capel, M.I., and Tomeu, A.J.: Parallel Cellular Automaton Tumor Growth Model. In: Practical Applications of Computational Biology and Bioinformatics, 12th International Conference, pp. 175–182 (2019)
12. Towns, J., Cockerill, T., Dahan, M., Foster, I., Gaither, K., Grimshaw, A., Hazlewood, V., Lathrop, S., Lifka, D., Peterson, G.D., *et al.*: XSEDE: accelerating scientific discovery. Computing in science & engineering 16(5), 62–74 (2014)

ICCS Camera Ready Version 2022
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-08751-6_1