Разработка пакета для моделирования деления клеток на графических ускорителях*

М.А. Кривов[1,2], Н.Ю. Захаров[1,2,] Ф.И. Атауллаханов[1,3], П.С. Иванов[1,2]

МГУ им. М.В. Ломоносова[1], ООО «ТТГ Лабс»[2], ЦТП ФХФ РАН[3]

Авторами был разработан программный пакет для моделирования митоза, позволяющий провести численные опыты на базе предложенной трёхмерной модели деления клетки с 29 изменяемыми параметрами. Ключевой чертой пакета является возможность одновременного осуществления расчётов для сотен независимых клеток с целью получения качественной статистики. Для обеспечения приемлемой скорости работы была подготовлена CUDA-версия алгоритма, которая в зависимости от моделируемого сценария обеспечивала 2 — 40 кратное ускорение по сравнению с одним ядром центрального процессора.

Ключевые слова: биофизика, митоз, деление клетки, численное моделирование, CUDA, GPU

1. Введение

Нарушения процесса деления клеток (митоза) у человека являются причиной различных патологий. Этот процесс, состоящих из нескольких фаз, сводится к удвоению числа хромосом и их последующему строго одинаковому (в норме) распределению между двумя дочерними клетками. Ключевую роль в митозе играет сочетание многочисленных ферментативных реакций и внутриклеточных физико-химических процессов, в которых участвуют сложные надмолекулярные комплексы. Фундаментальное место митоза в клеточном цикле и его роль в этиологии разных заболеваний делают крайне актуальной задачу моделирования митотических процессов и предсказания возможного влияния на их протекание низкомолекулярных лигандов.

Среди болезней, одним из направлений лечения которых является медикаментозное воздействие на процесс деления клеток, стоит особо отметить онкологические заболевания. Успехи в их лечении, достигнутые за последние 20 лет, в первую очередь обусловлены проведением фундаментальных исследований механизма деления клеток, в результате чего было открыто более 100 белков в составе кинетохора, которые могут быть выбраны в качестве мишеней для противоопухолевых препаратов [1].

На данный момент, по оценкам авторов, во всём мире изучением митоза занимается более 50 научных групп, однако все исследования преимущественно проводятся на молекулярном уровне и ориентированы на описание механизмов взаимодействия отдельных белковых структур, в то время как для объединения полученных результатов обычно используются теоретические и эмпирические модели, не всегда имеющие экспериментальное обоснование. В связи с этим одним из актуальных направлений исследований, к которому принадлежит и данная работа, является создание комплексной математической модели, которая является достаточно универсальной для того, чтобы описать процессы разных временных масштабов (длящихся от долей секунд до часов), и в которую можно включать новые блоки по мере развития исследований взаимодействия белковых комплексов.

В отличие от других работ, посвященных численному моделированию отдельных клеточных органелл и надмолекулярных структур, задействованных в митозе [2, 3], поставленная авторами цель заключается не в разовом получении конкретных результатов, а в создании универсального расчётного пакета, позволяющего многократно проводить моделирование рассматриваемых фаз деления клетки для различных сценариев при варьировании ключевых параметров.

* Исследование выполнено при финансовой поддержке РФФИ в рамках научного проекта № 16-07-01064а

2. Математическая модель

За основу была взята модель клетки из работы [4], но вместо двумерного случая рассматривался трёхмерный. Это, с одной стороны, позволило точнее воспроизвести исследуемые процессы, которые, очевидно, являются трёхмерными, но с другой, существенно увеличило объём вычислений, что обусловило необходимость распараллеливания вычислений.

Детальное описание, а тем более обоснование используемой биофизической модели митоза выходит за рамки данной статьи, поэтому ниже рассмотрены только основные этапы и дано общее описание рассматриваемого явления. Сразу стоит отметить, что в отличие от других предметных областей, таких как, например, гидродинамика или прочностной анализ, проверить полученные результаты экспериментальным путём практически невозможно. Диаметр клетки равен всего 10-20 мкм, а размер представляющей наибольший интерес структуры — кинетохора — составляет около 0.5 мкм. Из-за столь малых величин верификацию полученных результатов можно проводить только по косвенным данным, например, времени расцепления хромосом или скорости их движения.

В результате этого исходная задача заключается не столько в создании параллельной реализации конкретного алгоритма, сколько в создании инструмента, позволяющего итерационно дополнять модель новыми блоками, чтобы с её помощью можно было описать всё большее и большее число экспериментально установленных фактов. Так как каждая подобная проверка требует проведения порядка тысячи виртуальных опытов, то разработка параллельной версии вычислительного алгоритма должна вестись одновременно с его доработкой.

2.1. Моделируемые объекты

Согласно используемой в данный момент модели митоза, в прометафазе, метафазе и начале анафазы клетку можно схематично изобразить так, как показано на рис. 1.

Рис. 1. Визуализация виртуальной клетки для 40 (слева) и 1500 (справа) микротрубочек.

Все объекты аппроксимируются недеформируемыми геометрическими фигурами наиболее близкой к ним формы. Клетка представлена в виде сферы радиусом около 8 мкм, за пределы которой не может выходить ни один из объектов. Внутри клетки симметрично расположены два полюса деления, определённые как материальные точки. Из них под телесным углом в 180 градусов растут микротрубочки, суммарное число которых обычно варьируется от 1500 до 3000. Микротрубочка является белковой структурой, которая может либо удлиняться, либо укорачиваться, причём переход из одного состояния в другое с некоторой вероятностью происходит самопроизвольно.

Для описания хромосомы, на моделируемых этапах состоящей из двух идентичных сестринских хроматид, используется композиция из шести полуцилиндров (по три на каждую хроматиду), длина которой составляет около 5 мкм, а радиус — всего 0.5 мкм. Верхний и нижний цилиндры соответствуют плечам хромосомы, а средний задаёт кинетохор — область, за которую могут зацепляться микротрубочки.

Две сестринские хроматиды связаны между собой в районе центромеры, что не позволяет им разделиться до окончания метафазы. В рассматриваемой модели данное закрепление реализовано в виде либо небольшого абсолютно упругого стержня, либо, в зависимости от настроек, растягиваемой пружинки длиной около 0.01 мкм (рис. 2).

Рис. 2. Центромеры с мягким (слева) и жёстким (справа) закреплением.

Таким образом, в модели имеется пять типов объектов (клетка, полюса, микротрубочки, хромосомы, пружинки), для конфигурации которых введены 29 изменяемых параметров, отвечающих как за размеры, так и характер взаимодействия рассматриваемых объектов друг с другом.

3.2. Моделируемые взаимодействия

Основной интерес представляет динамика закреплений микротрубочек на кинетохоре, так как под действием создаваемых ими сил происходит движение хромосом и последующий разрыв пружинок, завершающий метафазу. В случае, если какая-либо часть микротрубочки соприкасается с кинетохором, с некоторой вероятностью с ней происходит зацепление, после чего хромосома начинает притягиваться к соответствующему полюсу деления.

Один из экспериментально установленных фактов заключается в том, что до момента разрыва связи сестринские хромосомы могут двигаться только в плоскости, которая перпендикулярна линии, соединяющей полюса деления. Это связано с тем, что полюса деления изначально находятся «над» хромосомами, и только после этого расходятся к противоположным сторонам клетки. Таким образом, на кинетохорах достаточно быстро образуется существенное число зацеплений, которых в дальнейшем оказывается достаточно для удержания хромосом в равновесном состоянии. В случае виртуального опыта проявления этой особенности добиться достаточно сложно из-за большого числа варьируемых параметров, поэтому на данном этапе развития модели имеется ряд искусственных ограничений на движение, которые применяются в зависимости от моделируемой фазы митоза.

Каждая нерасцеплённая пара хромосом имеет семь степеней свободы (положение, ориентация, длина пружинки) и испытывает два вида сил (притяжение к полюсам и диффузия). Если учесть, что из-за высокой вязкости цитоплазмы линейным и угловым ускорениями можно пренебречь, то для описания состояния отдельно взятой пары хромосом можно использовать следующую систему уравнений:

$$\alpha(t) - \gamma\vec{V} + \sum_{i=1}^{N_{MTs}} \vec{F}_i = 0$$

$$\beta(t) - \eta\vec{\omega} + \sum_{i=1}^{N_{MTs}} \vec{M}_i = 0$$

$$\left(\sum_{i=1}^{Nl} \vec{F}_i - \sum_{i=1}^{Nr} \vec{F}_i, \vec{n} \right) = \Delta l K$$

где $\vec{V}$ и $\vec{\omega}$ - линейная и угловая скорости пары сестринских хромосом, γ и η - коэффициенты поступательного и вращательного трения, $\vec{F}_i$ и $\vec{M}_i$ - сила и вращательный момент, возникающие под действием *i*-ой микротрубочки, Δl и K - растяжение и коэффициент упругости пружинки.

Расцепление всех пар хромосом происходит синхронно при достижении на каждой из них заданного настройками растяжения пружинки, после чего каждая хромосома становится независимым объектом с шестью степенями свободы и достаточно быстро притягивается к соответствующему полюсу. Координаты и углы поворота вычисляются исходя из значений скоростей путём их численного интегрирования с помощью метода Эйлера, а моделирование осуществляется с шагом 0.1 секунды для интервала продолжительностью в несколько часов.

3. Программная реализация

Несмотря на кажущуюся простоту алгоритма, объем реализующего его кода оказался достаточно большим — более 30 тыс. строк, что обусловлено несколькими причинами. Во-первых, результатом виртуального опыта является отнюдь не набор состояний клеток в разные моменты времени, а сделанные по ним усреднения, тип которых заранее неизвестен и определяется исходя из проверяемого факта. Это потребовало разработки системы плагинов, позволяющих обрабатывать первичные данные и получать требуемую статистику. Во-вторых, как показала практика, из-за технических причин типа недостаточности памяти, нехватки места на диске или отказа отдельных узлов кластера расчёты достаточно часто прерываются аварийно, и, как следствие, требуется восстановить сохранённые данные. Для этого был разработан собственный потоковый и весьма экономичный формат *.cell, который устойчив к ошибкам нескольких видов, и встраивание которого заметно сказалось на размере пакета.

3.1. Структура пакета

Логически пакет разделён на три уровня, для работы с каждым из которых требуется соответствующая подготовка исследователя. Первым из них является солвер, реализованный в виде консольного приложения в версиях для Windows и Linux, который в том числе может быть запущен на кластере. Данный солвер, получая на вход файл проекта, проводит требуемые расчёты и подготавливает так называемые траектории клеток — файлы в формате .cell, содержащие самодостаточное описание их состояний во все моменты времени.

Далее, для анализа полученных первичных данных исследователем должны быть подготовлены программы-анализаторы, собирающие интересующую его статистику и сохраняющие её в произвольном формате, обычно в виде CSV-таблиц. Для работы с траекториями клеток, полученными на предыдущем этапе, в рамках проекта были подготовлены библиотеки для языков программирования C++ и C#, позволяющие «открыть» cell-файлы и выгрузить все требуемые объекты. Предполагается, что подобным анализом в первую очередь будут заниматься специалисты по биофизике, поэтому C# был выбран как один из наиболее простых и понятных языков.

Наконец, последним, третьим уровнем являются утилиты, оформленные в виде графических приложений пользователя, для использования которых не требуются какие-либо знания программирования или опыт работы с командой строкой. На данный момент они представлены двумя программами — визуализатором, позволяющим просмотреть весь процесс деления клетки, и стандартным анализатором, собирающим базовую статистику по зацеплениям микротрубочек.

Стоит отметить, что уже имеется успешный опыт апробации текущей версии пакета несколькими сторонними командами, поэтому в дальнейшем планируется сохранить подобную структуру, развивая преимущественно сам солвер.

3.2. CUDA-версия алгоритма

Основной проблемой адаптации алгоритма под графические ускорители, как легко догадаться, стало существенное количество ветвлений и крайне небольшой параллелизм, что обусловлено спецификой проверок на соударение геометрических объектов. В результате этого нити из каждого полуварпа, работающие в SIMD-режиме, выполняли излишние действия, и итоговая производительность оказывалась достаточно низкой. Положение ещё больше усугублялось тем, что отдельные мультипроцессоры простаивали из-за недостаточности порождаемых нитей.

Решить данную проблему сугубо техническими приёмами практически невозможно, так как изменения должны быть внесены на уровне алгоритма, поэтому в данной работе была использована такая особенность предметной области, как необходимость проведения множества однотипных опытов. Действительно, если порождать один CUDA-поток на каждую микротрубочку, то всего будет выполняться около 1500-3000 нитей, а этого недостаточно, чтобы целиком задействовать несколько тысяч ядер ускорителя. Однако, если одновременно

моделировать 50 клеток, то объёма вычислений уже хватает для полной (хоть и не эффективной) загрузки всех мультипроцессоров, что дополнительно сглаживает проблему ветвления. При этом для хранения информации о состоянии одной клетки требуется около 100-200 килобайт, поэтому нехватки памяти ускорителя можно не опасаться.

Другим весьма неожиданным препятствием стала побайтовая воспроизводимость результатов. Одно из требований к пакету заключается в возможности внести для «дефектного» опыта корректировку (образно говоря, добавить в клетку «лекарство») и воспроизвести расчёты с тем же зерном генератора случайных чисел. В случае графического ускорителя достижению этого свойства препятствовали сразу две особенности технологии CUDA. Во-первых, из-за отличий в округлении последнего знака после запятой возникло несоответствие с исходной «золотой» версией алгоритма, для чего пришлось задействовать ассемблерные вставки. Во-вторых, из-за случайного порядка суммирования массивов (например, сил со стороны микротрубочек) наблюдалось накопление «погрешности», что было решено добавлением предварительной сортировки.

Ещё одним существенным моментом, потребовавшим внесения изменений в архитектуру пакета, стал формат данных. Как отмечалось ранее, все структуры изначально были оформлены в ООП-стиле в целях их удобного отображения в плагины и программы-анализаторы. Однако при использовании технологии CUDA все поля структур должны быть «перемешаны» таким образом, чтобы каждая нить адресовала последовательную ячейку памяти. Для этого пришлось перейти на двухуровневую модель, в которой все исходные данные хранятся в виде перемешенных массивов (так называемая концепция structure-of-arrays), выровненных по 64 байта, в то время как для работы с ними в ООП-стиле используются шаблонные классы-обёртки. Благодаря возможности перегрузки операторов и неявного приведения типов удалось полностью скрыть реальный формат данных, и, что более важно, подобное усложнение никак не сказалось на производительности эталонной версии алгоритма для центрального процессора — компилятор сумел распознать и убрать излишние конструкции и косвенные адресации.

Стоит отметить, что дополнительно были применены универсальные приёмы, подходящие для оптимизации большинства CUDA-программ, а именно — кэширование таблицы смещений в константной памяти, группировка запросов на копирование массивов по шине PCI-E в один, использование выделенной (page locked) памяти, сортировка промежуточных данных в разделяемой памяти с помощью атомарных операций, совместное использование текстурного и L1/L2 кэшей. К сожалению, объём статьи не позволяет подробно остановиться на каждом приёме и рассмотреть достигнутый с его помощью эффект.

4. Оценка производительности

Тестирование осуществлялось на системе, оснащённой центральным процессором Intel Xeon E3-1230 (4 ядра, частота 3.2 ГГЦ) и графическим ускорителем NVIDIA GeForce 780 Ti (2880 ядер, 980 МГц) в одинарной точности. На рис. 3 приведена зависимость нормированной производительности от параметров, напрямую задаваемых проверяемой гипотезой, а именно — от числа хромосом и микротрубочек.

Как легко заметить, перенос вычислений на ускорители оказывается оправданным только для «тяжёлых» сценариев, в которых имеется не менее 10 пар хромосом и 1500 микротрубочек,

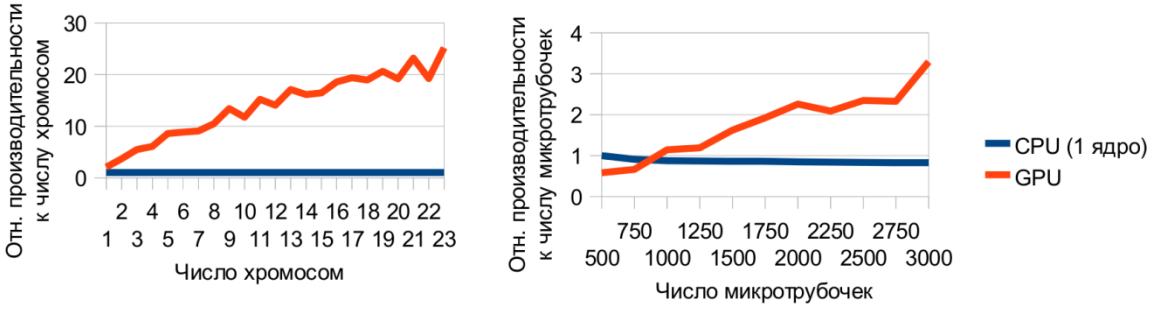

Рис. 3. Зависимость нормированной производительности от параметров клетки.

так как ускорение при этом доходит до 25 и даже до 40 раз. В остальных же случаях выигрыш практически отсутствует, что является основной проблемой текущей версии, для исправления которой требуется внесение изменений в сам алгоритм.

Другой тест заключался в варьировании параметров запуска, которые являются служебными и не влияют на получаемые результаты. В данном случае в роли подобных параметров выступали количество одновременно проводимых опытов и частота сохранения состояний клеток на диск, что показано на рис. 4.

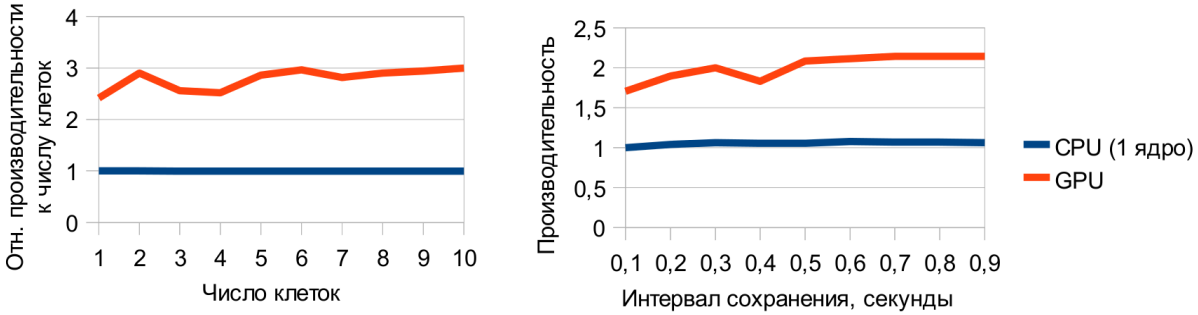

Рис. 4. Зависимость нормированной производительности от параметров запуска.

Если рассматривать представляющие наибольший интерес конфигурации клетки, то незначительный эффект от подобной подстройки заметен только в CUDA-версии солвера (22% и 26%), хотя в менее приближённых к реальности тестах ускорение доходит до 2-3 раз.

Резюмируя, стоит отметить, что выигрыш от переноса алгоритма на ускорители оказался локальным, а сопутствующие затраты — весьма существенными. Поэтому логично задаться вопросом, а был ли обоснован выбор именно данного типа вычислителей. Ответ заключается в том, что в рамках последующих этапов геометрические тела будут заменены на деформируемые сеточные объекты. Данный класс задач практически идеально подходит под архитектуру ускорителей, поэтому в будущем переход именно на них окажется оправданным.

5. Заключение

В статье описаны промежуточные результаты, полученные при реализации проекта по разработке комплексной математической модели деления клетки и её формализации в виде универсального расчётного пакета. В рамках текущего этапа была создана версия алгоритма для CUDA-совместимых графических ускорителей, благодаря которой удалось в разы повысить скорость расчётов. В частности, если раньше проверка одной гипотезы на группе из 50 клеток занимала порядка 43 часов, то теперь это время сократилось, в зависимости от рассматриваемого сценария, до 1-20 часов.

Дальнейшим направлением работ в первую очередь является разработка нового алгоритма с меньшим количеством ветвлений, что является крайне актуальным при расчётах с небольшим числом микротрубочек и хромосом. Другая приоритетная задача заключается в создании утилит для автоматизации запуска расчётов на кластере и проведении виртуальных опытов уже не на десятках, а на тысячах клеток.

Литература

1. Cheeseman I.M. (2014). The kinetochore. Cold Spring Harb. Perspect. Biol., vol. 6: a015826.
2. Loughlin R., Heald R., Nédélec F. (2010). A computational model predicts Xenopus meiotic spindle organization. J. Cell Biol., vol. 191, pp. 1239-1249.
3. Paul R., Wollman R., Silkworth W.T., Nardi I.K., Cimini D., Mogilner A. (2009). Computer simulations predict that chromosome movements and rotations accelerate mitotic spindle assembly without compromising accuracy. Proc. Nat. Ac. Sci. USA., v. 106, pp. 15708-15713.
4. Zaytsev A., Ataullakhanov F., Grishchuk E. (2013). Highly transient molecular interactions underlie the stability of kinetochore-microtubule attachment during cell division. *Cell. Mol. Bioeng.*, vol. 6, pp. 393-405.

Development of software for modeling cell division on graphics accelerators*

M.A. Krivov[1,2], N.U. Zakharov[1,2,] F.I. Ataullakhanov[1,3], P.S. Ivanov[1,2]

Lomonosov Moscow State University[1], TTG Labs LLC[2], CTP PCP of RAS[3]

The authors have developed a software package for mitosis simulation that allows to carry out numerical experiments based on a proposed three-dimensional model of cell division with 29 variable parameters. A key feature of this software is the possibility to perform simultaneous computations for hundreds of independent cells to obtain quality statistics. To ensure an affordable software performance, a CUDA version of the algorithm has been developed. Depending on the simulated scenario, this version provides a 2- to 40-fold computations speed up when compared to a single-core CPU.

Keywords: biophysics, mitosis, cell division, numerical simulation, CUDA, GPU

References

1. Cheeseman I.M. (2014). The kinetochore. Cold Spring Harb. Perspect. Biol., vol. 6: a015826.
2. Loughlin R., Heald R., Nédélec F. (2010). A computational model predicts Xenopus meiotic spindle organization. J. Cell Biol., vol. 191, pp. 1239-1249.
3. Paul R., Wollman R., Silkworth W.T., Nardi I.K., Cimini D., Mogilner A. (2009). Computer simulations predict that chromosome movements and rotations accelerate mitotic spindle assembly without compromising accuracy. Proc. Nat. Ac. Sci. USA., v. 106, pp. 15708-15713.
4. Zaytsev A., Ataullakhanov F., Grishchuk E. (2013). Highly transient molecular interactions underlie the stability of kinetochore-microtubule attachment during cell division. Cell. Mol. Bioeng., vol. 6, pp. 393-405.

* The reported study was funded by RFBR according to the research project № 16-07-01064a