РАЗРАБОТКА ПАРАЛЛЕЛЬНОГО АЛГОРИТМА КОМПЬЮТЕРНОГО МОДЕЛИРОВАНИЯ ВОДНО-ИОННОЙ ОБОЛОЧКИ ДНК.

Аветисян А.И., Гайсарян С.С., Калугин М.Д. (Институт Системного Программирования РАН, Москва), Теплухин А.В. (Институт Математических Проблем Биологии, Пущино, Московская Область)

DEVELOPMENT OF PARALLEL ALGORITHM FOR COMPUTER SIMULATION OF WATER-ION DNA COVER.

Avetisyan A.I., Gaissaryan S.S., Kalugin M.D. (Institute for System Programming RAS, Moscow), Teplukhin A.V. (Institute of Mathematical Problems of Biology RAS, Pushchino, Moscow region)

In this paper a parallel program for simulating water-ion cover of DNA fragments using Monte-Carlo method is presented. A cubic cell filled with water molecules and natrium and chloride ions is simulated. ParJava system, which allows instruments for developing parallel programs in Java language with MPI library, was used in our work.

Physical features of investigated system are calculated on statistically significant sample of molecular configuration in elementary cell, which is carried out with Monte-Carlo method in the classical scheme of Metropolis algorithm.

The process of computer simulation consists of 3 stages. On the first stage water cover is prepared with an independent parallel program. On the second stage a DNA fragment is placed in the center of a cell and Na^+ and Cl^- ions are inserted with the use of pseudorandom number generator. After finishing preparative stages, thermal motion of water molecules and ions around DNA fragment is simulated.

Results for simulating DNA fragment consisting of 150 base pairs (15 loops of duplex DNA, 9555 atoms) are presented also. Calculations were performed on ISP RAN cluster (12 nodes, 2 CPU Intel(R) Xeon(R) X5355 each, Myrinet-2000 interconnect). For simulation 80 cores of total 96 were used, each core handled 1 MPI-process. It took 9 hours to perform 10000 of experiments with each molecule in the system and first stage – preparation or water cover took about 3 hours.

Введение

Цель настоящей работы состоит в разработке масштабируемой параллельной программы для моделирования теплового движения молекул воды и ионов в присутствии фрагмента ДНК методом Монте-Карло. Экспериментальные методы исследования ДНК не могут дать достаточно точных результатов, кроме того, они являются чрезвычайно ресурсоёмкими. Поэтому в настоящее время компьютерное моделирование становится одним из мощнейших средств исследования биохимических процессов, происходящих в водной среде живых тканей.

Объем вычислений, необходимых для моделирования фрагмента ДНК, окруженного водной оболочкой, настолько велик, что возникает необходимость в выполнении программы на высокопроизводительных вычислительных системах [1]. Кластеры являются одним из распространенных типов таких систем. Ряд инструментов для разработки масштабируемых параллельных программ для кластеров реализован в среде *ParJava*[2], которая использовалась в данной работе.

Общая задача моделирования теплового движения воды разделена на три стадии, каждая из которых реализована в виде независимой параллельной программы. Первая программа занимается подготовкой водной конфигурации. Вторая загружает в сгенерированную водную оболочку фрагмент ДНК и ионы. Третья программа моделирует тепловое движение. Основная вычислительная нагрузка ложится на первую и третью программу, которые строят выборку конфигураций системы методом Монте-Карло по алгоритму Метрополиса, реализуемому по классической схеме [3]. Эти параллельные программы аналогичны с точки зрения распределения данных и структуры обменов.

Поэтому алгоритм будет рассмотрен на примере первой программы, которая занимается подготовкой водной оболочки.

Настоящая работа состоит из 6 разделов. В разделе 2 кратко описывается физическая модель моделируемой ячейки. В разделе 3 описана реализация параллельного алгоритма и модификации, которые были произведены над исходной программой. В 4 разделе описана структура обменов данными параллельной программы. В 5 разделе приведены графики масштабируемости и сравниваются исходный и модифицированные алгоритмы. В разделе 6 приведены результаты моделирования водно-ионной оболочки фрагмента В-ДНК.

Физическая модель

Расчеты методом Монте-Карло различных характеристик воды выполняются в данной модели в ходе компьютерных экспериментов с системами кубической формы. При этом система координат связывается с одной из троек ребер куба, имеющих общую вершину. Для устранения поверхностных эффектов и уменьшения зависимости результатов от размера системы на ячейку налагают периодические граничные условия вдоль координатных осей. Потенциальная энергия такой системы представляет собой сумму энергий парных взаимодействий молекул внутри ячейки, а также их взаимодействий с образами этих молекул, заполняющими соседние ячейки пространства. Количество вычислений можно существенно уменьшить, используя метод ближайшего образа, который основан на том, что энергия взаимодействия молекул воды быстро убывает с ростом расстояния между ними. Таким образом, это позволяет отказаться от вычисления энергии взаимодействия *i*-й молекулы со всеми (26+1) образами *j*-й, оставив только один – ближайший.

Энергия взаимодействия двух молекул воды предполагается равной сумме энергий взаимодействий трех атомов одной молекулы с тремя атомами другой. При этом энергия взаимодействия атома i одной молекулы с атомом j другой вычисляется по формуле Леннарда-Джонса [4]:

$$E_{ij}(r_{ij}) = q_i q_j / r_{ij} - A_{ij} / r_{ij}^m + B_{ij} / r_{ij}^{12} \qquad (1)$$

Здесь r_{ij} – расстояние между атомами, q_i и q_j – их заряды, A_{ij} и B_{ij} – параметры потенциала. Величина m равна 10 для пар атомов О и Н или 6 для пар ОО и НН. Значения этих параметров, а также геометрические характеристики молекулы воды описаны в работе [1]. Если расстояние между молекулами (ближайшие образы) превосходит некоторую величину (например, $R_c = 8\,\text{Å}$), энергию их взаимодействия можно обнулить не вычисляя. Такой прием называют обрезанием потенциала. В этом случае, размер ребра элементарной ячейки не должен быть меньше R_c

Физические характеристики изучаемой системы рассчитываются по статистически значимой выборке молекулярных конфигураций элементарной ячейки, осуществляемой методом Монте-Карло по алгоритму Метрополиса, реализуемому по классической схеме. Температура системы, размер ячейки и число молекул воды в ней в процессе моделирования не изменяются (*NVT*-ансамбль). Исходное положение молекул может быть произвольным, но в пределах ячейки. Практически моделирование происходит следующим образом. При заданной температуре T (в нашей работе $T = 300K$) случайное изменение положения молекулы воды (элементарное испытание), приводящее к изменению энергии системы от E_1 к E_2, принимается, если $E_2 \le E_1$, или, в противном случае, если $\exp\left[\frac{E_1 - E_2}{kT}\right] > \xi$, где ξ – случайное число, равномерно распределенное в интервале $(0,1)$, а k – постоянная Больцмана. Если пробное состояние не принято, в выборку еще раз включается прежняя конфигурация молекул воды.

Элементарное испытание состоит из смещения молекулы воды на $\delta(1-2\alpha_i)\,\text{Å}$ и ее поворота вокруг случайно выбранной оси, проходящей через центр атома кислорода, на угол

$\gamma(1-2\beta)$ радиан. Здесь α_i и β − случайные числа, равномерно распределенные в интервале $(0,1)$, а i − x-, y- и z- координаты атомов данной молекулы. Величины δ и γ выбираются так, чтобы в ходе испытаний отвергалось не более половины новых конфигураций. Для водных систем при нормальных внешних условиях $\delta = 0.125\,\text{Å}$, $\gamma = 0.125$ радиан. Если в результате смещения молекула покидает ячейку, то, в соответствии с периодическими граничными условиями, она возвращается с противоположной стороны.

Полученная в результате многократного повторения этой процедуры совокупность конфигураций образует (в пределе $N\to\infty$, N − число испытаний) канонический ансамбль Гиббса (в нашем случае − *NVT*-ансамбль), где вероятность выбора конфигурации с энергией E пропорциональна больцмановскому фактору $exp(-E/kT)$, а переходы между конфигурациями обладают свойствами цепей Маркова.

Параллельный алгоритм

Параллельная программа разработана в среде *ParJava*[2] на языке *Java* с использованием библиотеки *MPI*. Программа моделирует тепловое движение молекул воды методом Монте-Карло с использованием процедуры Метрополиса.

Молекулы воды в данной модели описываются трехмерными декартовыми координатами составляющих их атомов и углами поворота относительно координатных осей. Исходный куб делится на домены, имеющие форму параллелепипеда. Разбиение определяется числом доступных процессоров. Данные, необходимые для описания молекул, содержащихся в доменах, распределяются по процессорам в соответствии с их декартовыми координатами. При этом учитываются следующие ограничения:

- По каждому из направлений исходная ячейка должна быть разбита не менее чем на 2 части;
- Размеры параллелепипеда, по длине, ширине и высоте не могут быть меньше $2R_c + 2\delta$.

На каждый процессор может приходиться несколько десятков тысяч молекул в зависимости от размеров системы.

Для корректных расчетов в данной модели требуется хранить образы молекул, содержащихся в соседних доменах и вспомогательную информацию, необходимую для определения положения молекулы внутри домена и расположения домена в ячейке.

На одной итерации все процессоры производят испытания с каждой из распределенных на них молекул. Для того чтобы результаты испытаний в одном домене не оказывали влияния на соседние домены, они разделяются на 8 равных по объему поддоменов, как показано на рис. 1.

Размер домена должен быть больше 2 радиусов обрезания по каждому из измерений. Таким образом, размер каждого из поддоменов по длине, ширине и высоте будет больше радиуса обрезания. Это гарантирует, что молекулы, которые попадают в радиус обрезания и учитываются при расчете энергии взаимодействия, могут находиться только в том же поддомене, либо в прилегающих. Списком соседей мы будем называть молекулы, которые учитываются при вычислении изменения энергии.

80-85% времени работы программа тратит на вычисление изменения потенциальной энергии, произошедшей вследствие перемещения молекулы. Для расчета изменения энергии необходимо вычислить энергию взаимодействия молекул до, и после испытания. Список соседних молекул строится при каждом испытании и не сохраняется до следующего испытания. Сначала вычисляется расстояние между молекулами в декартовых координатах. Если оно меньше радиуса обрезания, то молекула попадает в список соседей и сразу рассчитывается энергия их взаимодействия по формуле Леннарда-Джонса для старого и нового положения, в противном случае программа переходит к расчету расстояния до следующей по номеру молекулы.

В программе можно выделить две операции, на которые приходится основной объем вычислений. Для построения списка соседей необходимо вычислять квадрат расстояния

между молекулами в декартовых координатах. А для молекул, удовлетворяющих условиям необходимо вычислить энергию взаимодействия по формуле Леннарда-Джонса.

Оценим сложность этих двух операций. Вычисление квадрата расстояния требует 3 операций умножения и 8 операций сложения. Вычисление энергии взаимодействия двух молекул воды по формуле Леннарда-Джонса состоят из нескольких сотен операций сложения и умножения, и требует, в 55-60 раз больше времени на выполнение.

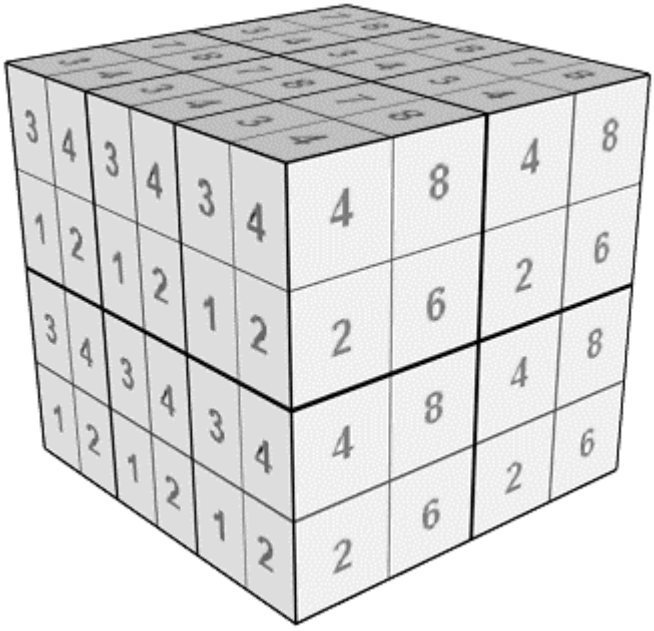

Рисунок 1. Разбиение исходного объема на домены и поддомены. Толстыми линиями отделяются домены, расположенные на разных процессорах, тонкими линиями отделены поддомены. Цифры означают порядок обхода поддоменов на каждом процессоре.

Выбор соседей производится перебором всех молекул в поддомене, которому принадлежит испытуемая молекула и прилегающих к нему. Каждый из поддоменов содержит примерно одинаковое число молекул. Эти молекулы можно разделить на 2 категории, те молекулы, вклад которых учитывается при вычислении изменения энергии и те, вкладом которых можно пренебречь. В исходном алгоритме расстояние от испытуемой молекулы до каждой молекулы из всех девяти поддоменов вычисляется дважды – для старого и нового положения (рис 2.). Если учесть то, что используемой модели молекула может изменить свое положение не больше, чем на δ, причем $\delta << R_c$, большинство молекул, не дававших вклада в энергию старой конфигурации, не будут давать вклада и в новой конфигурации. При этом расстояние до каждой из них будет вычисляться дважды.

На рисунке 2 приведен случай, когда одна из сторон домена равна $2R_c+ 2\delta$, то есть на каждый процессор приходится минимальное количество молекул (~250). При условии, что каждый домен делится на 8 равных частей, не всегда можно добиться такого распределения, вследствие больших размеров задачи и ограниченного числа процессоров.

Оценим для трехмерного случая отношение числа молекул, дающих вклад в вычисление энергии к общему числу молекул в прилегающих доменах. Поскольку молекулы воды равномерно распределены в пространстве, то количество молекул, попадающих в список соседей, относится к общему числу молекул, как объем сферы с радиусом R_c относится к суммарному объему прилегающих поддоменов. Для вычисления изменения энергии, также требуется вычислить энергию системы для нового положения молекулы. Так как оно отличается от прежнего не более чем на δ, оценим число молекул величиной $k\frac{4}{3}\pi(R_c+\delta)^3$, где k – количество молекул в единице объема. Допустим, размеры домена в точности равны $2R_c+ 2\delta$. В таком случае общее количество молекул в соседних поддоменах оценивается, как $k(3R_c+3\delta)^3$. После сокращения получится, что число молекул

участвующих в вычислениях составляет $\frac{4/3\pi}{27}$, что приблизительно равно 15,5% процента. Если же величина домена увеличится в 2 раза, то количество молекул, учитываемых при вычислении энергии, составит 1,9%. Операция вычисления расстояния в декартовых координатах имеет большую вычислительную сложность, поэтому потребовалась модификация исходного алгоритма для уменьшения объема вычислений.

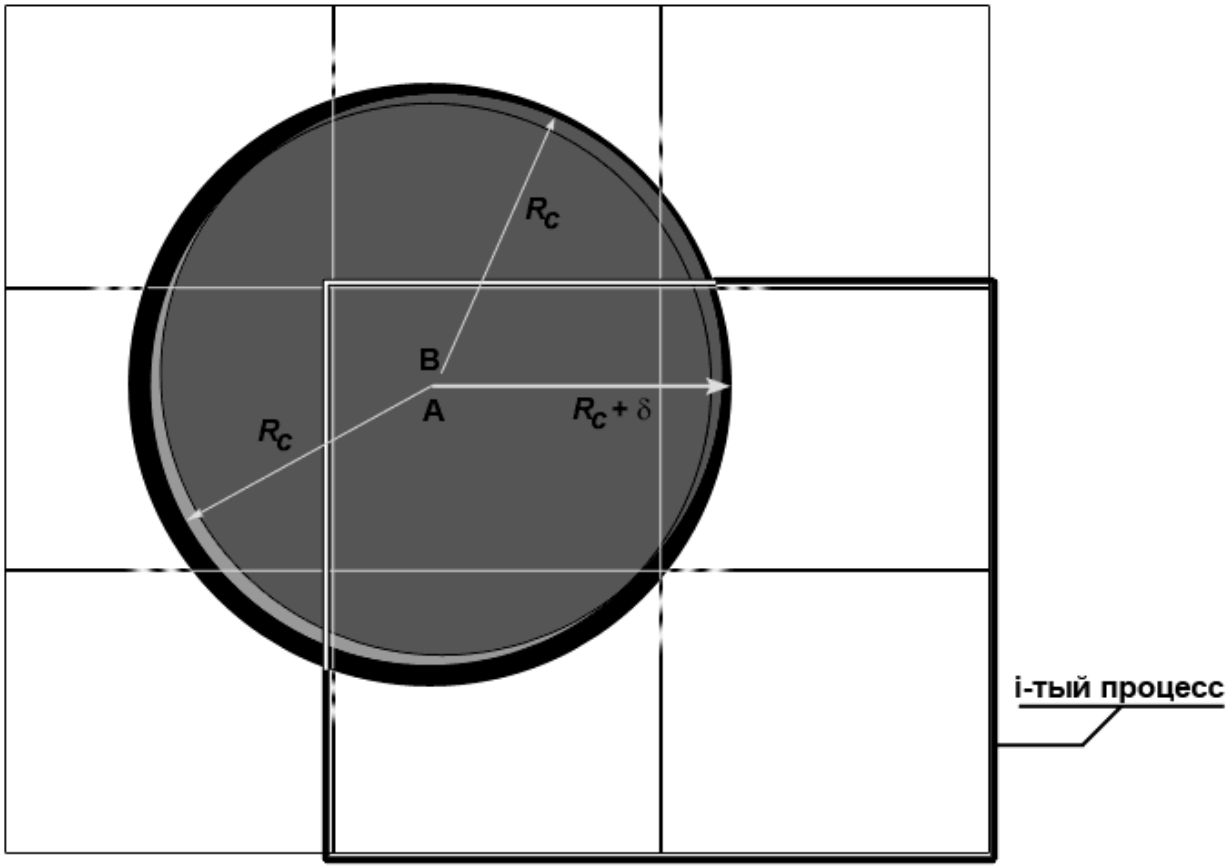

Рисунок 2. Вычисление изменения энергии, двумерный случай. Прямоугольники показывают разбиение области на поддомены, жирной линией показан i-тый домен, относящийся к i-тому процессу. Точки A и B означают старое и новое положение испытуемой молекулы. Оттенками серого закрашены области, попадающие в радиус обрезания R_c для старого и нового положения. Черным цветом показана окружность радиуса $R_c + \delta$, с центром в точке A.

Из рисунка 2 видно, что площадь окружности, с радиусом $R_c + \delta$ центр которой совпадает со старым положением молекулы, не намного больше площади окружности радиуса R_c. При этом все молекулы, которые попадают в радиус обрезания для старого и нового положения молекулы, также попадут и в сферу с радиусом $R_c + \delta$. Воспользовавшись этим можно уменьшить объем вычислений, не изменяя результат.

При вычислении изменения энергии сначала строится список и вычисляется энергия взаимодействия соседей для старого положения молекулы (на рис. 2 обозначено буквой **A**). Если при построении списка соседей для некоторой молекулы расстояние до **A** больше $R_c + \delta$, то молекула заведомо не войдет в список соседей и для нового положения (на рис.2 обозначенного буквой **B**), поэтому расстояние между молекулой и положением **B** вычислять не требуется, оно будет заведомо больше радиуса обрезания. Если же расстояние оказывается меньше $R_c + \delta$, расчет производится как в исходном алгоритме. Описанная модификация позволяет сократить количество операций по вычислению расстояния только для нового положения молекулы.

На объем вычислений влияет также разбиение домена на поддомены. В случае, показанном на рис. 2 разбиение является оптимальным, размеры поддоменов близки к величине радиуса обрезания. При этом на каждый процессор приходится минимально

возможный объем данных. Если увеличить размеры доменов, то объем данных на процессоре увеличится, при этом возрастает объем вычислений при каждом расчете изменения энергии в элементарном испытании. Это происходит потому, что количество молекул, попадающих в смежные поддомены, увеличивается и для построения списка соседей требуется перебрать и вычислить расстояние до большего числа молекул.

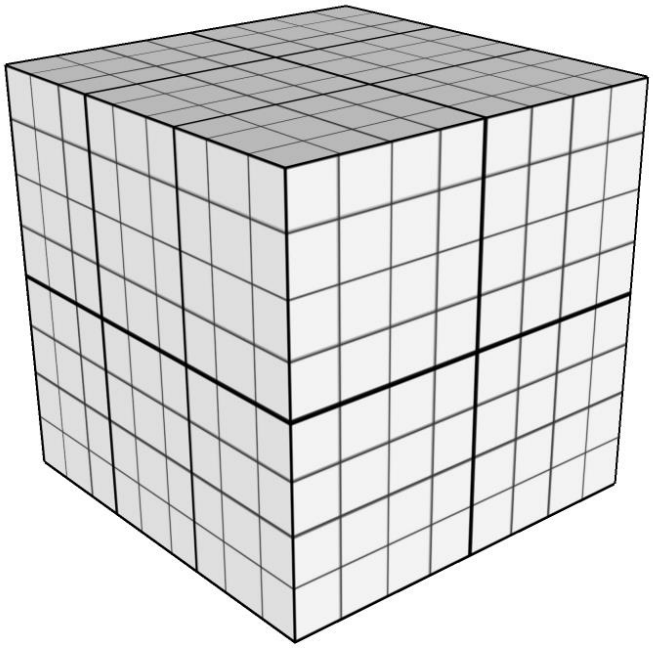

Рисунок 3. Новое разбиение ячейки на домены и поддомены. Толстыми линиями отделены домены, тонкими линиями – поддомены.

При моделировании больших объектов, размеры доменов, рассчитываемых одним процессором, могут быть в несколько раз больше радиуса обрезания. В исходном алгоритме домен делился на 8 равных частей (рис. 1), размер которых увеличивался с увеличением размеров домена. Для того чтобы уменьшить объем перебора и уменьшить количество вычислений при построении списка соседей, было введено новое разбиение на поддомены. Каждый домен теперь делится на минимально возможные поддомены, с условием, чтобы каждый поддомен по длине ширине и высоте был не меньше радиуса обрезания (рис. 3), поэтому по каждому из измерений размер поддомена $R_c + \delta < x_i < 2R_c$. Таким образом мы гарантируем, что для построение списка соседей надо обойти только поддомен в котором находится испытуемая молекула и прилегающие к нему. Расчеты производились при $R_c = 8{,}55\,\text{Å}$, то есть в худшем варианте сторона поддомена имеет длину не больше $17{,}1\,\text{Å}$, а в переборе участвует не более 4300 молекул. На рисунке 3 изображено разбиение исходной ячейки на 12 доменов, при этом каждый домен разбит на 48 поддоменов.

Структура обменов данными в программе

По алгоритму Метрополиса после каждого испытания вычисляется изменение энергии и новая конфигурация либо принимается, либо отбрасывается. В соответствии с данными экспериментов над реальными системами, параметры датчика случайных чисел настроены так, что принимается чуть больше 40% новых конфигураций. Для получения конфигураций, имеющих характеристики, отвечающие экспериментальным данным, требуется несколько тысяч итераций (не менее 10000).

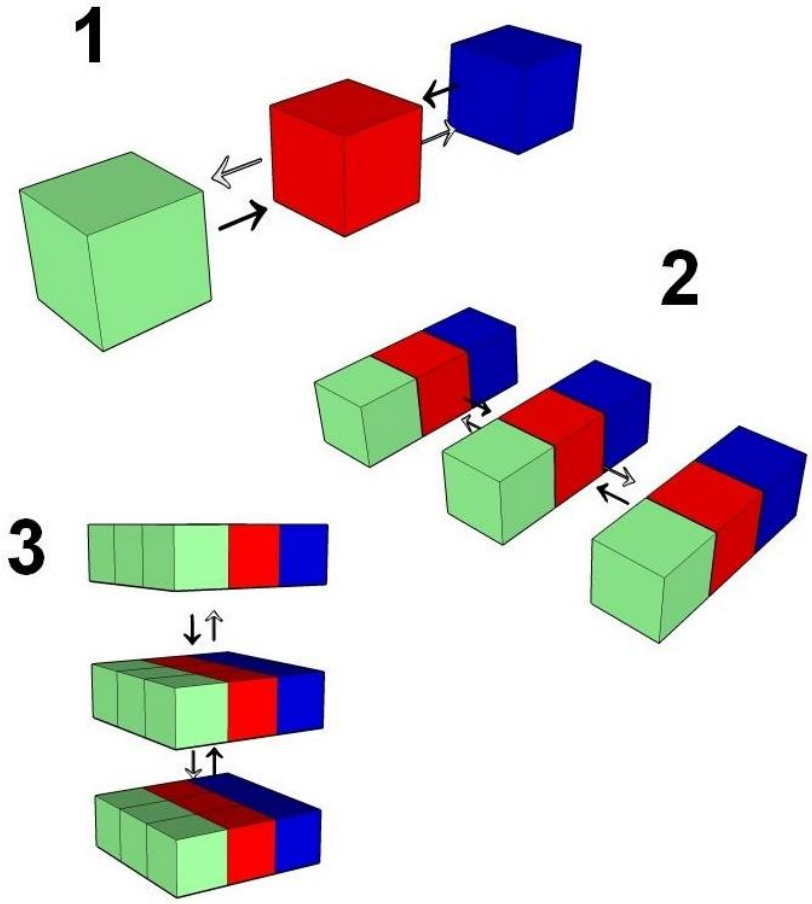

Рисунок 4. Этапы коммуникаций. На первом шаге процессоры обмениваются координатами атомов своей области, с процессорами в направлениях +x и –x. На втором шаге процессор обменивается 3 наборами координат, которыми он владеет с процессорами в направлении +y и –y. На каждом процессоре есть 9 наборов координат, которые передаются процессорам в направлении +z и –z на третьем шаге.

Обход частей домена осуществляется всеми процессами в одинаковом порядке, и после обхода каждого поддомена процессы обмениваются данными. В результате испытаний часть молекул изменяет свое положение, а некоторые молекулы оказываются в соседних доменах. Это приводит к тому, что образ, который хранится на процессе, обрабатывающем соседний домен, отличается от оригинала. Поэтому перед проведением следующей серии испытаний необходимо произвести обмен данными между процессорами и обновить образы.

В библиотеке *MPI* реализована декартова топология, которая в точности соответствует примененному разбиению на домены и позволяет упростить структуру обменов. Данные между процессами передаются с использованием функции *MPI_SENDRECV*. Схема обменов, использованная в программе, предложена в работе [5]. Процессоры производят обмены, накапливая данные, полученные от соседей (рис.4).

На первом шаге процессоры передают пересчитанные координаты молекул в направлении координатной оси X (рис. 4.1). После этого, полученные от соседних процессоров данные, вместе с результатами собственных вычислений передаются в направлении координатной оси Y, после чего все накопленные данные пересылаются в направлении координатной оси Z. Такая схема позволяет за 3 пары обменов обновить данные обо всех прилегающих параллелепипедах.

Для контроля состояния системы необходимо через определенное количество итераций вычислять полную энергию ячейки. Для этого вычисляется потенциальная энергия молекул расположенных на одном процессе. Полученные значения суммируются с помощью функции *MPI_REDUCE*.

Для проведения длительных расчетов больших систем в программе реализован механизм точек останова. После выполнения заданного количества итераций программа сохраняет на жесткий диск текущие координаты молекул ячейки и вспомогательную информацию, необходимую для продолжения моделирования с того места, на котором вычисления были остановлены.

Результаты сравнения исходного и модифицированного алгоритмов.

Для оценки эффективности произведенных модификаций мы сравнили 3 программы. Исходная программа на языке Фортран с использованием *MPI*, сравнивалась с двумя программами на языке *Java*, также использующими *MPI*. Первая программа на *Java* включает в себя только одну модификацию с введением дополнительного радиуса обрезания $R_c + \delta$, во второй программе также модифицирован метод разбиения на домены, описанным выше способом. Исходная программа на Фортран допускала запуск только на числе процессоров, равному кубу натурального числа, то есть на 8, 27, 64 процессорах. Программы на *Java* допускают выполнение на произвольном числе процессоров, с одним ограничением, по каждому из направлений исходная ячейка должна делиться хотя бы на 2 домена.

Для сравнения производительности была проведена серия тестов с разным числом процессов. Объем задачи увеличивался с ростом числа процессоров таким образом, чтобы на каждом процессоре был одинаковый объем данных. На 8 процессах моделировалась кубическая ячейка с ребром 100 ангстрем, на 64 процессах ребро ячейки составляло 200 ангстрем, при этом каждый процесс производил моделирование для 4166 молекул. В процессе моделирования над каждой из молекул проводилось по 1000 испытаний.

Результаты измерения, приведенные на рисунке 5 получены на кластере ИСП РАН, состоящем из 12 узлов (*2 x Intel Xeon X5355*, 4 ядра), объединенных сетью *Myrinet*. Число *MPI*-процессов соответствовало числу ядер, каждое из которых может рассматриваться как отдельный процессор. Для программы на Фортран измерения производились с использованием 8, 27 и 64 ядер. Для программ на *Java* измерения производились с использованием 8, 12, 16, 24, 27, 32, 36, 40, 48, 56, 64 ядер.

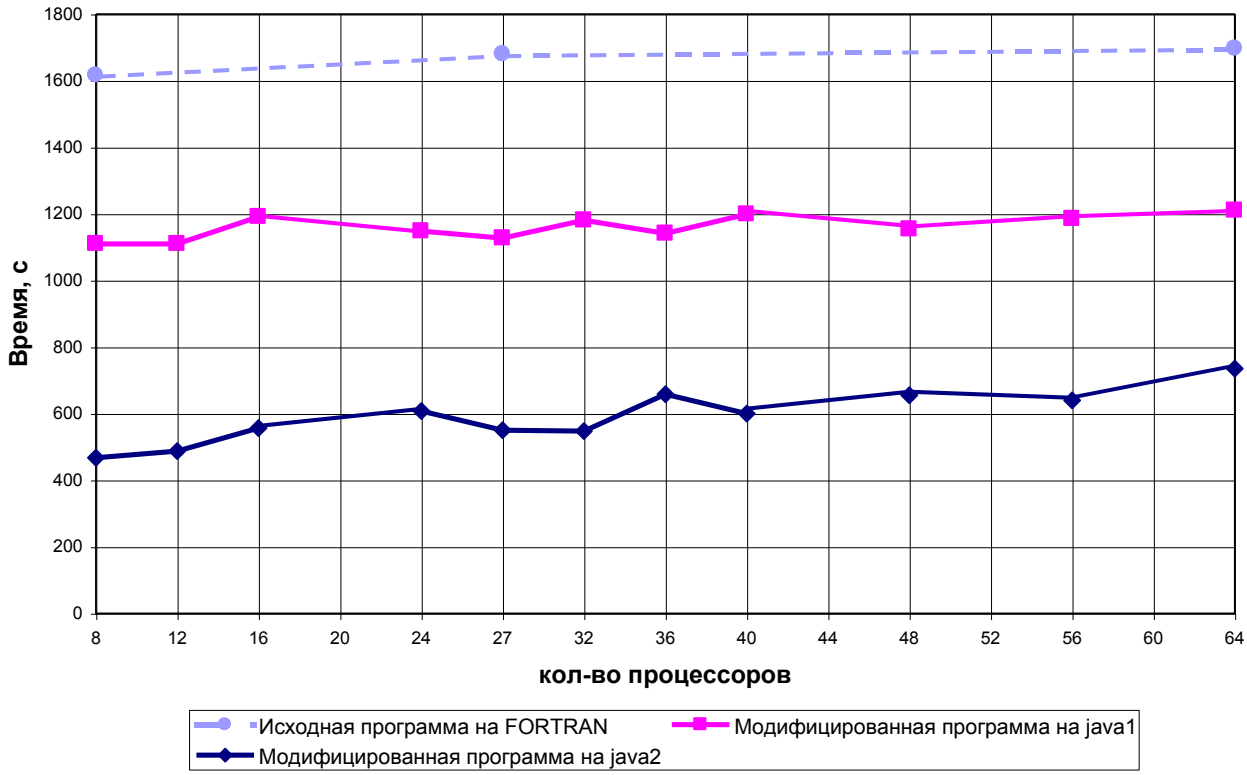

Рисунок 5. Сравнение исходной программы на языке *FORTRAN77* с модифицированными программами на языке *Java*.

Из графика видно, что при равных объемах данных первая модифицированная программа на *Java* работает в 1,5 раза быстрее, а вторая в 2-3 раза. Программа на *Java*, в

точности соответствующая исходной программе на языке Фортран, требовала от 10 до 30% больше времени. Использование среды *ParJava* позволило обнаружить области в программе, которые могут быть модифицированы для ускорения ее работы. Следует отметить, что на графике приведены результаты работы разных алгоритмов, реализованных на языках Фортран и *Java*. То есть нами не ставилась цель сравнить скорость работы одной программы на разных языках, более того, если аналогичные модификации реализовать в исходной программе на Фортран, то она будет работать быстрее аналогичной программы на языке *Java*. При этом язык *Java* позволяет производить модификации с меньшими затратами усилий программиста и быстрее отлаживать новые версии программ.

Обращает также внимание то, что время работы второй модифицированной программы возрастает с увеличением числа процессоров. Это связано с тем, что при существующей схеме обменов увеличивается объем накладных расходов, связанных с передачей и приемом сообщения, а также с ростом количества неявных синхронизаций. Дальнейшая оптимизация параллельной программы будет связана с изменением схемы коммуникаций для уменьшения накладных расходов.

Результаты моделирования водно-ионной оболочки.

Параллельная программа, моделирующая движение воды вокруг молекулы ДНК, использует результаты моделирования теплового движения воды. В ячейку с равномерно распределенными молекулами воды, полученную в результате работы соответствующей программы, помещается фрагмент ДНК, при этом молекулы воды, находящиеся ближе некоторого расстояния от фрагмента удаляются. Размеры ячейки выбираются так, чтобы расстояние от фрагмента ДНК до границ ячейки составляло $10-20\,\text{Å}$.

Любую биомолекулу можно представить в виде совокупности компактных и, по возможности, электронейтральных фрагментов, содержащих небольшое число атомов. Образец исследуемой ДНК разбивается на нуклеотиды, а в каждом из них выделяется фосфатная группа, остаток дезоксирибозы и азотистое основание, определяющее тип соответствующего нуклеотида. При таком способе разбиения получаются фрагменты из 5-15 атомов, расположенные в пределах сферы радиуса $R_b = 2-3\,\text{Å}$. Для каждого фрагмента определяются координаты его геометрического центра, и, в соответствие с их значениями, программа генерации начальных данных распределяет фрагменты по доменам.

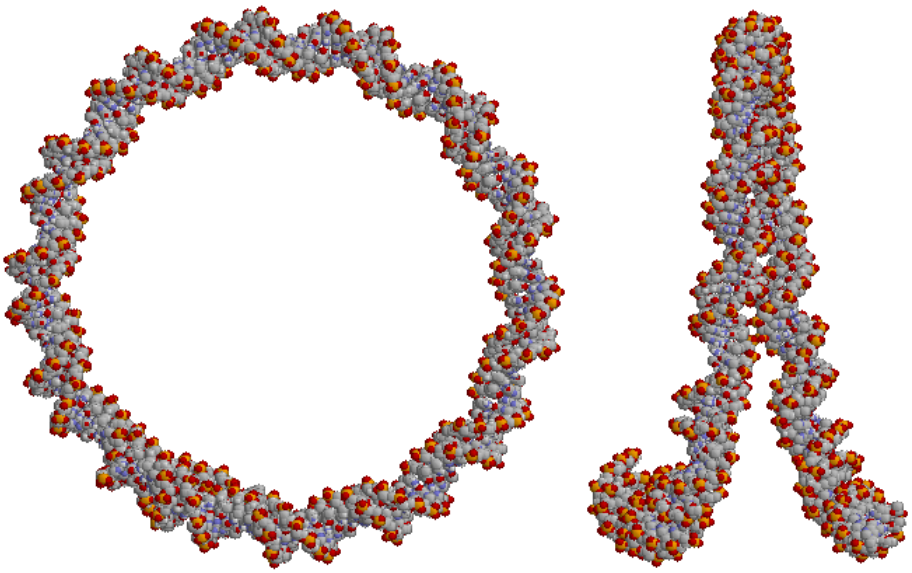

Рисунок 6. Фрагмент B-ДНК, использованный при тестовом моделировании (две проекции).

После того как фрагмент ДНК помещен в ячейку, производится моделирование теплового движения воды вокруг него. Принцип моделирования схож с моделированием движения молекул воды. Проводится элементарное испытание, в результате которого

молекула воды изменяет свое положение, после чего вычисляется изменение энергии, вызванное этим перемещением. Отличие состоит в том, что при вычислении энергии взаимодействия молекул в этой программе учитывается взаимодействие молекул воды с группами атомов, составляющими фрагмент ДНК. Для вычисления энергии взаимодействия, также используется потенциал Леннарда-Джонса, при этом учитываются взаимодействия на больших расстояниях, чем в случае взаимодействия двух молекул воды.

Программа для моделирования теплового движения ионов в растворе использует те же начальные данные, что и программа, моделирующая движение воды в присутствии фрагмента ДНК. Моделирование производится по аналогичному алгоритму, отличия заключаются в части вычисления энергии взаимодействий в системе, поскольку необходимо учитывать 2 новых вида взаимодействия вода – ион и ион – ион. Взаимодействие вида ион – ион имеет большую дальность действия, чем взаимодействия вода – вода и вода – ион, что необходимо учитывать при разбиении данных по процессам.

Кроме того, реализована возможность помещения фрагмента ДНК в раствор ионов в воде и моделирования водно-ионной оболочки ДНК. Для тестирования программ моделировался фрагмент B-ДНК, состоящий из 150 пар нуклеотидов (15 витков двойной спирали, 9555 атомов) и водная оболочка, содержащая ионы Cl^- и Na^+. Размер ячейки, содержавшей фрагмент $220\,\mathring{A}$ и содержал ~300 тысяч молекул воды.

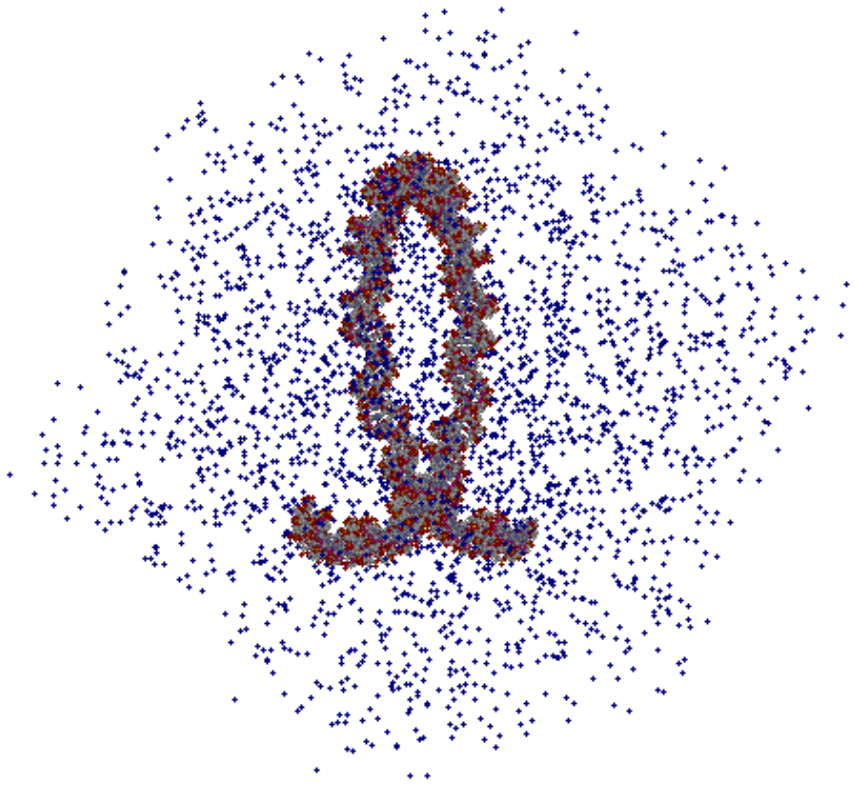

Рисунок 7. Распределение ионов Na^+ и Cl^- вокруг фрагмента ДНК.

ЛИТЕРАТУРА

1. Теплухин А. В. Многопроцессорное моделирование гидратации мезоскопических фрагментов ДНК. // Математическое моделирование, 2004г., том 16, номер 11, с.15-24.
2. Иванников В.П., Гайсарян С.С., Аветисян А.И., Падарян В.А. Оценка динамических характеристик параллельной программы на модели. // Программирование, 2006г., том 32, номер 4, с. 203-215.

3. Metropolis N.A., Rosenbluth A.W., Rosenbluth M.N., Teller A.H., Teller E. Equation of state calculations by fast computing machines. // J. Chem. Phys., 1953, v.21, p.1087-1092.
4. Вуд В. Исследование моделей простых жидкостей методом Монте-Карло. // Физика простых жидкостей. Ред. Г. Темперли и др. − М.: Мир, т.2, 1973, с.275-394.
5. Heffelfinger G.S. Parallel atomistic simulations. // Comput. Phys. Commun. 2000, v.128, P. 219-237.