ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ   
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«СИБИРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ТЕЛЕКОММУНИКАЦИЙ И ИНФОРМАТИКИ»

**КУРСОВОЙ ПРОЕКТ**

по дисциплине “Параллельные вычислительные технологии”

на тему

**Разработка параллельной программы моделирования методом молекулярной динамики**

|  |  |
| --- | --- |
| Выполнил студент | Куликов Д.А. |
|  | Ф.И.О. |

|  |  |
| --- | --- |
| Группы | ИС-142 |
|  |  |

|  |  |  |
| --- | --- | --- |
| Работу принял |  | профессор д.т.н. М.Г. Курносов |
|  | подпись |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Защищена |  | Оценка |  |
|  |  |  |  |

Новосибирск – 2023

Оглавление

[ВВЕДЕНИЕ 3](#_Toc154408637)

[1. Теория 4](#_Toc154408638)

[1.1. Leapfrog 4](#_Toc154408639)

[1.2. Вопрос граничных условий (boundary conditions) 4](#_Toc154408640)

[1.3. Cell Subdivision 5](#_Toc154408641)

[1.4. Начальное состояние системы 5](#_Toc154408642)

[2. Анализ алгоритмов. 6](#_Toc154408643)

[2.1. Анализ памяти. 6](#_Toc154408644)

[2.2. Погрешность вычислений 6](#_Toc154408645)

[2.3. Вычислительная сложность 6](#_Toc154408646)

[3. Разбор реализованной программы. 7](#_Toc154408647)

[3.1. Основная программа 7](#_Toc154408648)

[3.2. Визуализация данных 8](#_Toc154408649)

[4. Анализ данных 9](#_Toc154408650)

[4.1. Визуальный анализ 9](#_Toc154408651)

[4.2. Характеристики системы 9](#_Toc154408652)

[4.3. Анализ алгоритма 10](#_Toc154408653)

[ЗАКЛЮЧЕНИЕ 11](#_Toc154408654)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 12](#_Toc154408655)

# ВВЕДЕНИЕ

Цель работы: решить задачу двухчастичных взаимодействий (two-body interactions) используя Leapfrog. В системе будут находиться частицы воздуха, то есть мягкие сферы (soft-disk fluid). Частицы изначально должны быть равномерно распределены. Скорости тел в начальный момент времени присваиваются случайные направления, а их фиксированная величина определяется температурой. При этом корректируются так, чтобы центр масс системы находился в покое, что устраняет общее движение системы. Метод вычисления взаимодействий между частицами разбиение ячеек (Cell Subdivision), каждый процесс работает со своей ячейкой.

Общий тип задач, где рассматриваются взаимодействия между N-частицами, известен как задача N-тел (N-body problem). В контексте молекулярной динамики (MD), где изучается движение и взаимодействие атомов или частиц, основная задача часто формулируется как задача двухчастичных взаимодействий (two-body interactions).

# Теория

## Leapfrog

Leapfrog - это метод численной интеграции, где решаются дифференциальные уравнения движения. В методе Leapfrog интегрирование производится "сдвигами" (leaps) переменных состояния системы через фиксированные промежутки времени.

Leapfrog часто используется в молекулярной динамике, потому что он является численно стабильным и обеспечивает хорошее сохранение энергии. Метод прост в реализации и поддержке. Предпочтителен для простых систем и задач с ограниченными ресурсами. Leapfrog использует мало сообщений между процессами.

Метод назван "leapfrog" из-за того, что координаты и скорости оцениваются в разные моменты времени. Чтобы оценка скорости соответствовала моменту времени, в который определяются координаты, то можно использовать формулу:

## Вопрос граничных условий (boundary conditions)

Возможные реализации границ:

* Конечные система: контейнер ограничен стенами, с которой будут сталкиваться частицы.
* Периодические граничные условия: в этой реализации нет стенок, которые влияли бы на поведение частиц. При выходе частиц из границ контейнера, возвращаются в него, но уже с обратной стороны.
* Бесконечная система: у системы нет границ, что облегчает вычисления.

В конечной и периодической системах приводится выполнять лишние вычисления расстояния частиц до стенок, при чём большая часть находится в дали от неё. От выбранной формы пространства зависит вычислительная сложность алгоритма. В трех измерениях легче вычисляется столкновение с прямоугольной или призматической областью. А использование более сложных форм области, такие как сфера или гексагон, увеличит отношение объема к поверхности и уменьшить влияние конечных размеров системы.

В моей программе будет использоваться периодическое пространство в форме куба.

## Cell Subdivision

Разбиение пространства на ячейки нужно для ускорения вычисления. Пространство однородно, поэтому если каждый процесс будет работать только с одной группой частиц, то больших изменений не произойдёт.

При использовании Cell Subdivision приходится использовать дополнительную память, для сохранения частиц, которые покинули границы своего контейнера. Так же усложняется процесс передачи сообщений между процессами, но взамен размер сообщений уменьшается.

## Начальное состояние системы

Основной идеей является то, что, проведя достаточно продолжительную симуляцию, мы должны получить результаты, не зависящие от выбора начального состояния. Для этого частицы должны быть равномерно распределены.

Скорости атомов в начальный момент времени присваиваются случайные направления, а их фиксированная величина определяется температурой. При этом корректируются так, чтобы центр масс системы находился в покое, что устраняет общее движение системы.

# Анализ алгоритмов.

## Анализ памяти.

Данных частиц хранятся в структуре с массивом масс и массивами векторов (3 измерения) позиции, скорости, силы.

Структура имеет размер: (3 \* 3 + 1) \* n \* sizeof(data\_type). Программа используются переменные типа double.

Если процессу будет доступен 1 Гб оперативной памяти, тогда приближённое значение максимального число частиц в системе можно вычислить:  
= = = 13 421 772.

Ранее уже упоминалось, что при разбиении пространства на контейнеры нужно использовать дополнительную память. И это нужно учитывать.

Если заранее известно, что система будет стабильна и частицы распределены равномерно. Можно сэкономить память выделив на каждом процессе память для частиц.

## Погрешность вычислений

Локальные ошибки, внесенные на каждом временном шаге, составляют порядок O() для координат и O() для скоростей.

## Вычислительная сложность

Аналогом Cell Subdivision является метод All-pairs, в котором вычисляется взаимодействие всех со всеми. Но All-pairs имеет сложность O(), а Cell Subdivision понижает O(*N*).

# Разбор реализованной программы.

## Основная программа

После инициализации MPI начинается разбиение пространства. Сначала вызывается функция MPI\_Dims\_create, которая разбивает процессы на трёхмерную решетку. Далее создаётся коммуникатор для этой сетки при помощи MPI\_Cart\_create. Для коммуникатора указываем, что пространство переодическое. Чтобы процессы получили свои координат вызывается MPI\_Cart\_coords. MPI\_Cart\_shift вызывается для поиска соседей, во время вычислений процессы будут обмениваться сообщениями только с соседями.

Число частиц между процессами распределяется равномерно, и каждый процесс самостоятельно инициирует их значения. В алгоритме инициации используется функция rand, чтобы каждый процесс генерировал разные значения вызывается srand(rank). Частицы изначально располагаются случайным образом, при большом числе тел алгоритм обеспечивает желаемую плотность. Скорости атомов в начальный момент времени присваиваются случайные направления, а их фиксированная величина определяется температурой. При этом корректируются так, чтобы центр масс системы находился в покое, что устраняет общее движение системы.

После инициации данных начинается работа цикла, в котором вызываются функции для вычисления перемещения, изменения скорости (SingleStep) и обмена сообщения (Exchange).

Ранее упоминалась формула, по которой работает leapfrog и исходя из неё LeapfrogStep должен быть разбит на две части. Одна вычисляет изменение скорости и перемещение, а вторая только изменение скорости. Между LeapfrogStep(1, n) и LeapfrogStep(2, n) вызывается CalculateForces(n).

В вычислении силы используется потенциал Леннарда-Джонса:

Чтобы избавиться от артефактов при вычислении силы используется отсечение и сглаживание. При очень близком расстоянии тела могли бы оказывать друг на друга большую силу. Сглаживание уменьшает силу, что так же устраняет ситуации, когда частица приобретает аномальную скорость.

Перед обменом сообщениями, нужно найти частицы, вылетевшие за границу. Для этого вызывается FindEscapees. Если частица вылетела за границу основного пространства, то она возвращается в него с обратной стороны. Данные о частицах хранятся структуре из массивов, но, чтобы передать частицы соседям, нужно перенести их данные в массив структур. Для удобной передачи таких структур создаётся специальный тип данных MPI.

В функции Exchange процессы передают соседям частицы, которые перешли в их контейнер. Для отправки используется MPI\_Isend. Перед приёмом вызывается MPI\_Probe и MPI\_Get\_count, чтобы узнать сколько частиц нужно принять. Затем вызывается MPI\_Irecv. После обмена, новые частицы вставляются на место частиц, которые покинули данный процесс. Если процесс принял частиц больше, чем отправил, то оставшиеся частицы добавляются в конец. Если процесс больше отправил, что лишние частицы удаляются.

Параллельно ведётся наблюдение за состоянием системы. Раз в stepWrite итераций каждый процесс записывается положение его частиц в файл, раз stepAvg итераций частицы передают в корневой процесс дополнительные данные при помощи MPI\_Reduce. Корневой процесс записывает в файл: число итераций, время, скорость всей системы, общая энергия, кинетическая энергия и давение.

## Визуализация данных

Для визуализации используется библиотека GLFW. При помощи этой библиотеке реализовано создание окна, графика и управление. Функция InitializeGLFW создаёт окно, инициализирует камеру, и подключает функцию, отвечающую за управление. Затем компилируются вершинный и фрагментный шейдеры. В шейдерах цвет сфер зависит от их массы.

Для отрисовки нужно считать положение частиц с файлов и отсортировать по удалённости от камеры. Эту задачу выполняет поток, созданный при помощи openMP. А за отрисовку должен отвечать поток №0, так как OpenGL плохо работает в многопоточных программах и с другими потоками GLFW работать не будет.

# Анализ данных

## Визуальный анализ

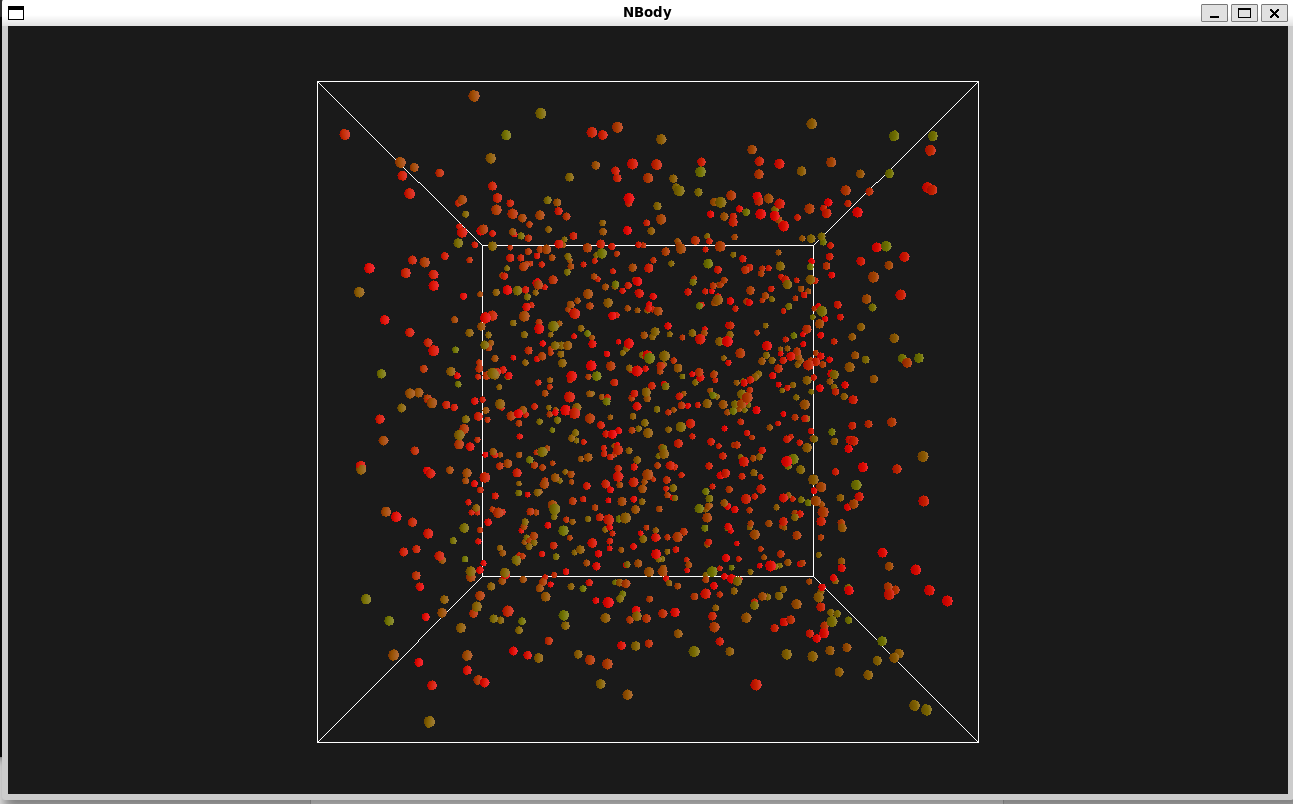


Рис. 1. Визуализация моделирования молекулярной динамики.

Визуализация помогает проверить корректность и позволяет наблюдать за лучше изучить динамику процессов. По полученным данным можно сказать, что программа работает корректно и система стабильна.

## Характеристики системы

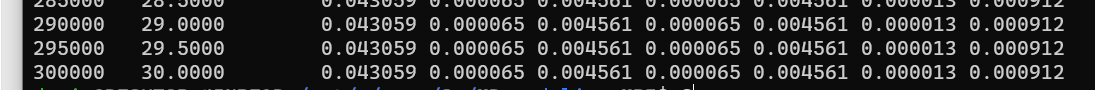
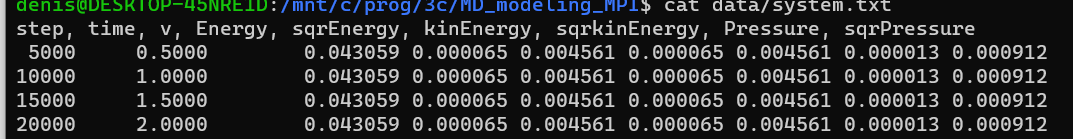


Рис. 2. Характеристики системы.

Характеристики системы подтверждают, суждение о том, что система стабильна. Закон сохранения энергии удовлетворён.

## Анализ алгоритма

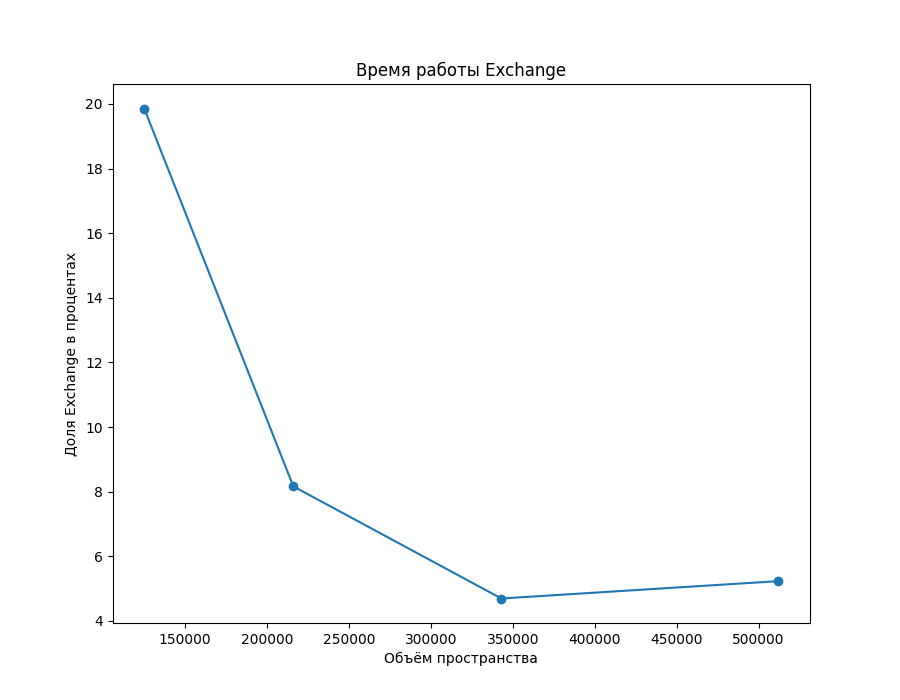


Рис. 3. Зависимость длительности обмена данных от объёма пространства.

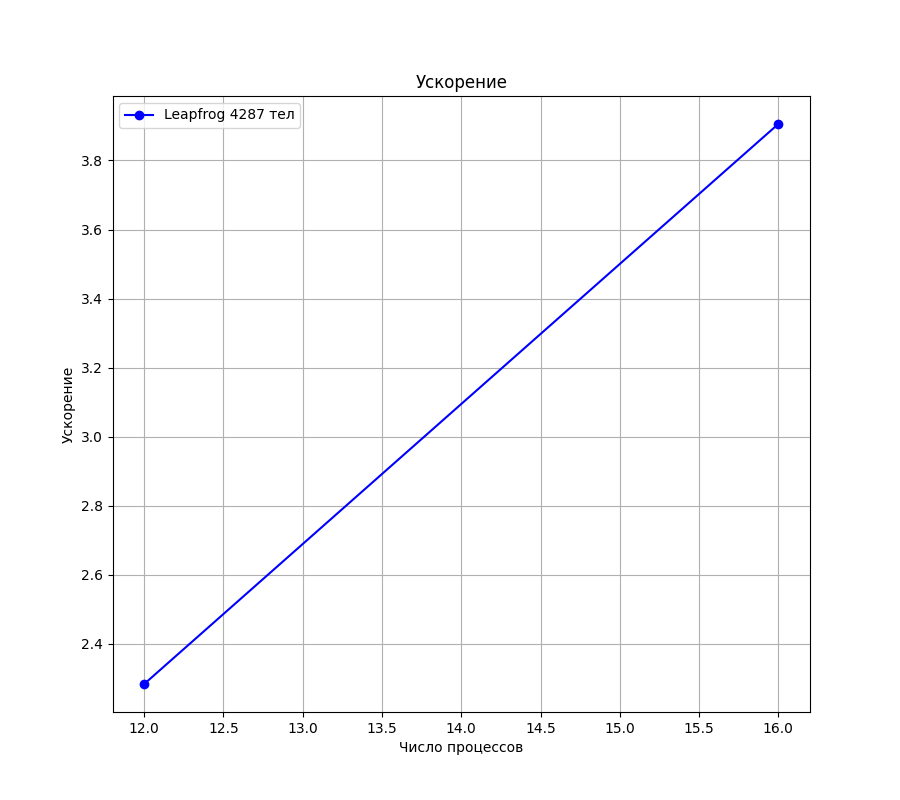


Рис. 4. Ускорение от числа процессов относительно .

# ЗАКЛЮЧЕНИЕ

В результате выполнения работы разработана параллельная программы моделирования молекулярной динамики метод численной интеграции, Leapfrog, с использованием Cell Subdivision.

В результате моделирования получилась корректная и стабильная система.

Алгоритм не требователен к памяти, имеет удовлетворительную точность и хорошо масштабируется.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. THE ART OF MOLECULAR DYNAMICS SIMULATION Second Edition D. C. RAPAPORT. – Cambridge University Press 1995, Dennis Rapaport 2004

http://www.r-5.org/files/books/computers/algo-list/fuzzy-logic/modeling/Dennis\_Rapaport-The\_Art\_of\_Molecular\_Dynamics\_Simulation-EN.pdf