ЛАБОРАТОРИЯ СИНТЕЗА ПАРАЛЛЕЛЬНЫХ ПРОГРАММ

Зав. лабораторией д.т.н. Малышкин В. Э.

Важнейшие достижения

Специализированные алгоритмы автоматического конструирования и исполнения высокопроизводительных параллельных программ для задач линейной алгебры с разреженными матрицами в системе LuNA

Беляев Н. А.

Разработаны специализированные алгоритмы конструирования и исполнения параллельных программ для системы LuNA, которые обеспечивают более высокую эффективность исполнения LuNA-программ для задач разреженной линейной алгебры. Ускорение достигается за счет того, что входная LuNA-программа проверяется на принадлежность к упомянутому классу задач, и если это так, то дальнейшая трансляция и исполнение программы осуществляются с использованием разработанных специализированных системных алгоритмов. Специализированная поддержка отдельных классов прикладных алгоритмов является стандартным способом повышения эффективности программ в системах параллельного программирования.

Основная идея разработанного решения базируется на распространенных в ручном параллельном программировании практиках реализации задач разреженной линейной алгебры. А именно, для множества вычислительных операций и переменных (фрагментов данных и вычислений LuNA-программы) формируется статически определенный граф информационно зависимых задач. Далее выполняется декомпозиция этого графа на множество подграфов по количеству вычислительных узлов. При разбиении минимизируются количество межузловых дуг и разница в количестве вершин на узел. Такое разбиение является известной задачей и решается с помощью пакета METIS. Далее на каждом вычислительном узле работает специализированная исполнительная система, выполняющая задачи в локальном графе готовности входных данных асинхронно в многопоточном режиме и асинхронно передающая необходимые данные соседним узлам.

На основе этого подхода была автоматически сконструирована тестовая параллельная программа описания на языке LuNA численного алгоритма матрично-векторного умножения разреженной матрицы на вектор. Производительность автоматически сконструированной программы лишь незначительно уступила реализации этого численного алгоритма из библиотеки Intel MKL

а *б*

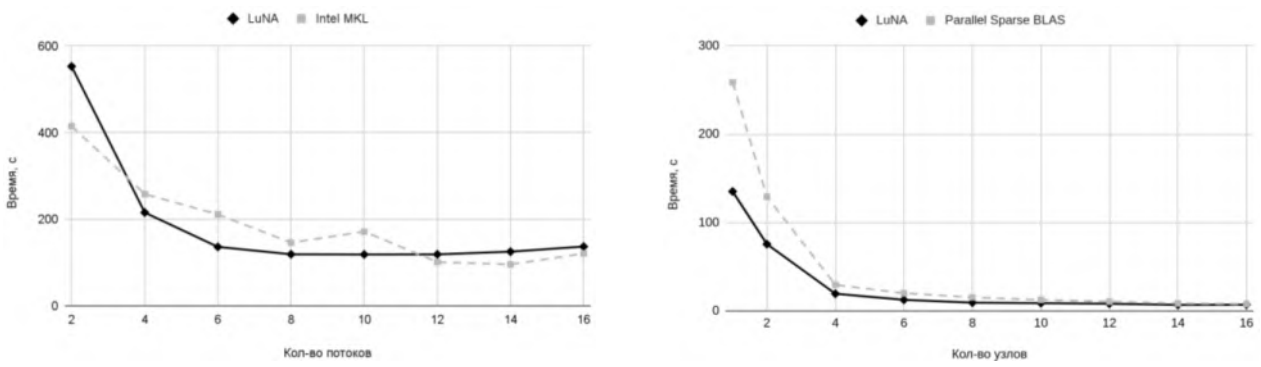


Рис. 1: Сравнение производительности параллельной программы, автоматически сконструированной системой LuNA с библиотечной реализацией параллельного умножения разреженной матрицы на вектор в составе библиотек Intel MKL (*а*) и Parallel Sparse BLAS (*б*)

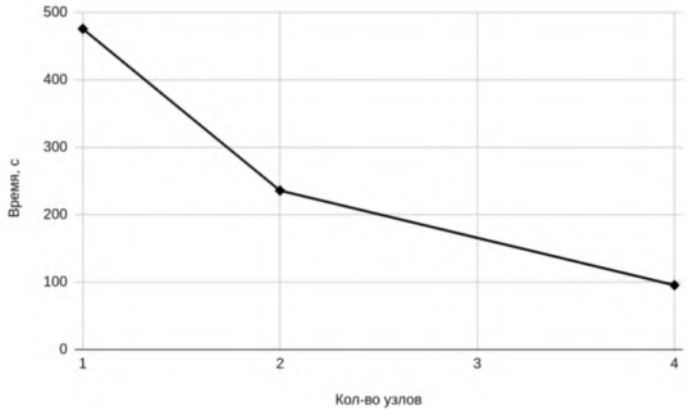


Рис. 2: Результаты измерения производительности параллельной программы, сконструированной системой LuNA по описанию алгоритма Гаусса –Зейделя предобусловливания разреженной СЛАУ

и превзошла производительность реализации из библиотеки Parallel Sparse BLAS (рис. 1), что является хорошим результатом в области автоматической генерации программ. Необходимо отметить, что разработка такой LuNA-программы существенно менее трудоемка, чем ручное программирование. Также на основе предложенного подхода реализован алгоритм предобусловливателя Гаусса – Зейделя для разреженной СЛАУ как пример задачи, не имеющей готового библиотечного решения, но попадающей в класс алгоритмов, реализуемых в рамках предложенного подхода. Тестирование показало высокий коэффициент распараллеливания сконструированной программы (рис. 2).

Результаты исследований опубликованы в работе

Беляев Н. А. Автоматическое конструирование высокопроизводительных параллельных программ для задач разреженной линейной алгебры в системе LuNA // Пробл. информ. 2022. № 3. С. 46–60. DOI: 10.24412/2073-0667-2022-3-46-60.

Отчет по этапам научно-исследовательских работ, завершенным в 2022 г. в соответствии с планом НИР института

Проект НИР № 1 "Суперкомпьютерные технологии решения больших задач естествознания, математические модели, методы анализа и оптимизации сложных информационных систем".

Номер государственной регистрации НИР 0251-2021-0005.

Руководители: д.ф-м.н. Марченко М. А., к.ф-м.н. Черных И. Г.

Раздел 1 "Высокопроизводительные вычисления. Создание методов, алгоритмов, инструментальных средств и пакетов прикладных программ для вычислительных систем сверхвысокой производительности".

Руководитель – д.т.н Малышкин В. Э.

Этап 2022 г. Блок 2 "Разработка экспериментальных модулей и функциональных баз активных знаний для применения на суперкомпьютерах; реализация класса задач моделирования физической среды с использованием библиотеки клеточно-автоматных топологий и решение тестовых задач моделирования".

1. Разработана подсистема автоматизированного выбора алгоритма динамической балансировки нагрузки на вычислительные узлы в системе LuNA.

В научном численном моделировании на суперЭВМ часто возникает проблема статического или динамического обеспечения баланса вычислительной нагрузки. Эта проблема не имеет

эффективного универсального решения, вследствие чего на практике используются различные частные и эвристические алгоритмы балансировки нагрузки на вычислительные узлы. Несмотря на то, что эта тема хорошо разработана в литературе и имеется большое количество методов, алгоритмов и программ балансировки нагрузки, их применение в каждом конкретном случае представляет собой проблему. Даже настройка параметров подходящего алгоритма балансировки нагрузки может стать непреодолимым препятствием для пользователя суперЭВМ. Это обусловливает актуальность автоматического обеспечения балансировки нагрузки на узлы как подзадачи автоматического конструирования параллельных программ. Если в системе программирования имеется набор алгоритмов балансировки в виде, допускающем их автоматическое применение, то обозначенная проблема становится неактуальной для пользователя. В системе автоматического конструирования параллельных программ LuNA имеются средства для накопления и автоматического применения алгоритмов статической и динамической балансировки вычислительной нагрузки на узлы.

В рамках системы LuNA разработана модульная подсистема автоматизированного выбора алгоритма статической или динамической балансировки вычислительной нагрузки. Подсистема подразумевает наличие нескольких алгоритмов динамической балансировки нагрузки, возможно имеющих параметры (например, допустимое значение дисбаланса), а также наличие модуля выбора действующего алгоритма балансировки и его параметров (метабалансировщика). На текущем этапе выбор осуществляется вручную, а метабалансировщик реализует его. В дальнейшем планируется автоматический выбор. Метабалансировщик является слабосвязанным заменяемым модулем системы, а модульная организация алгоритмов балансировки допускает добавление в систему новых алгоритмов динамической балансировки нагрузки (расширяемость). Три имеющихся в системе LuNA алгоритма балансировки нагрузки (один статический и два динамических) были оформлены в качестве таких модулей.

Для проверки работоспособности предложенной схемы на примере различных задач было выполнено экспериментальное исследование зависимость времени выполнения задач от алгоритма балансировки нагрузки и его параметров. Тестирование производилось на кластере Информационно-вычислительного центра Новосибирского государственного университета. Характеристика каждого узла: 12 ядер и 24 Гб ОЗУ.

Результаты экспериментального исследования приведены в табл. 1. Они носят предварительный характер, однако подтверждают, что в различных условиях предпочтительными оказываются разные алгоритмы динамической балансировки нагрузки или значения параметров этих алгоритмов. Система LuNA может выступать в качестве основы для накопления различных алгоритмов динамической балансировки нагрузки и экспериментального исследования подходов к автоматизации выбора этих алгоритмов из числа имеющихся и настройки их параметров.

2. Разработан инструментарий отладки фрагментированных программ на основе подхода "посмертный анализ" (luna_trace). Созданный инструмент способен обнаруживать ряд часто встречающихся семантических ошибок в программах для системы LuNA.

При использовании системы LuNA для конструирования численных параллельных программ существует проблема отладки. Отчасти она решается с использованием существующих средств и подходов, но для языка программирования LuNA, на котором пользователи составляют программы в соответствующей системе, характерны собственные логические ошибки, не имеющие

Таблица 1: Время выполнения программы в зависимости от алгоритма балансировки нагрузки и его параметров (NB – без балансировки, ST – статическая балансировка, WR – алгоритм Work Requesting, RB – алгоритм Rope of Beads)

Тип	Кол-во узлов	Параметры балансировки	Результат, сек
NB	1		128.88
WR	2	Количество соседей – 2	129.02
WR	4	Количество соседей – 2	97.94
WR	4	Количество соседей – 4	93.54
ST	2		129.92
ST	4		88.47
ST+WR	2	Количество соседей – 2	130.19
ST+WR	4	Количество соседей – 2	89.32
ST+WR	4	Количество соседей – 4	117.53
RB	2	Подотрезков – 20	130.59
RB	2	Подотрезков – 60	137.63
RB	4	Подотрезков – 20	91.57
RB	4	Подотрезков – 60	93.40
RB+WR	2	Подотрезков – 20; количество соседей – 2	128.76
RB+WR	2	Подотрезков – 60; количество соседей – 2	130.23
RB+WR	4	Подотрезков – 20; количество соседей – 2	92.86
RB+WR	4	Подотрезков – 20; количество соседей – 4	95.27
RB+WR	4	Подотрезков – 60; количество соседей – 2	93.17
RB+WR	4	Подотрезков – 60; количество соседей – 4	97.73

взаимно однозначного соответствия с ошибками в императивных языках, равно как и в параллельных программах, использующих MPI, OpenMP или другие распространенные технологии. В связи с этим требуются специфичные для фрагментированных программ средства отладки.

Компилятор языка LuNA в результате серии преобразований выдает на выходе код на языке C++, который далее можно транслировать каким-либо известным компилятором (например, gcc). Поскольку для программ на C++ создано множество разнообразных инструментов отладки, то можно пытаться отлаживать данное промежуточное представление в виде C++-кода. Но поиск ошибок в таком C++-коде традиционными средствами не решает проблему отладки полностью. Причина заключается в том, что пользователю трудно сопоставить данную C++-программу с исходной LuNA-программой. В этой связи известные диалоговые отладчики (gdb, TotalView, Arm DDT) не подходят для задачи отладки фрагментированных программ. К тому же в случае "больших" вычислительных программ предпочтительно автоматизированное средство отладки, выдающее список найденных ошибок без участия пользователя или с минимальным участием.

К настоящему времени создано инструментальное средство "посмертного анализа" luna_trace, при котором во время выполнения LuNA-программы журналируется и затем анализируется информация о ходе исполнения программы. Средство способно обнаруживать ошибки:

– множественной инициализации одного фрагмента данных несколькими фрагментами вычислений;

– отсутствия инициализации фрагмента данных, являющегося входным для какого-либо фрагмента вычислений;

– циклической зависимости по данным между фрагментами вычислений.

```
Following CFs appear to be root cause of the system hang:

set(b, a * a) [./root.fa:11] never finished
main
 in sub main() [./root.fa:9]

 Awaited DF a
 full name a, a declared in sub main
 - - - -

2 more CFs nrever finished but had data dependences on other hang CFs
```

Рис. 3: Вывод инструментального средства luna_trace

Кроме того, инструментальное средство обнаруживает факт зависания фрагментированной программы и автоматически прерывает ее в таком случае.

Программное средство написано на языке программирования Python и предоставляет пользователю интерфейс командной строки. Оно предназначено для анализа файлов трассы, генерируемых при выполнении фрагментированных программ исполнительной системой LuNA, с целью поиска некоторых типичных семантических ошибок и анализа их причин.

По ходу работы параллельного MPI-приложения, в которое в итоге транслируется LuNA-программа, на каждом процессе собирается трасса с информацией о запускаемых фрагментах вычислений, а также об обрабатываемых ими входных и выходных фрагментах данных. Созданное программное средство производит автоматическую агрегацию этих трасс и выводит пользователю сообщение об ошибке с исчерпывающей информацией о месте и причине (рис. 3).

Средство автоматизированного обнаружения семантических ошибок luna_trace может быть использовано в учреждениях науки, университетах и организациях, тематика которых связана с разработкой параллельных программ численного моделирования.

3. Разработана фрагментированная реализация метода частиц-в-ячейках (PIC) с использованием библиотеки управления распределенными данными Didal (disitributed data library). По результатам тестирования на вычислительном кластере с распределенной памятью производительность полученной фрагментированной реализации оказалась выше, чем у существующей MPI-реализации, при этом масштабируемость обеих реализаций оказалась сравнимой.

Библиотека управления распределенными данными Didal предназначена для упрощения разработки фрагментированных программ для вычислительных систем с распределенной памятью. С использованием библиотеки была разработана фрагментированная реализация PIC (particle-in-cell) метода для задачи моделирования динамики самогравитирующего пылевого облака. На рис. 4 показан результат работы программы (данные визуализированы с помощью визуализатора Paraview).

Тестирование производительности полученной фрагментированной реализации было проведено на кластере МВС-10П ОП2 МСЦ РАН (2 процессора Intel Xeon Gold 6248R (Cascade Lake) на вычислительный узел, коммуникационная сеть Intel Omni-Path). Для сравнения использовалась аналогичная MPI-реализация этой же задачи. В первом варианте (рис. 5) использовался фиксированный размер задачи со следующими параметрами: сетка 128^3 ячеек, 10^7 частиц, 8^3 фраг-

а *б*

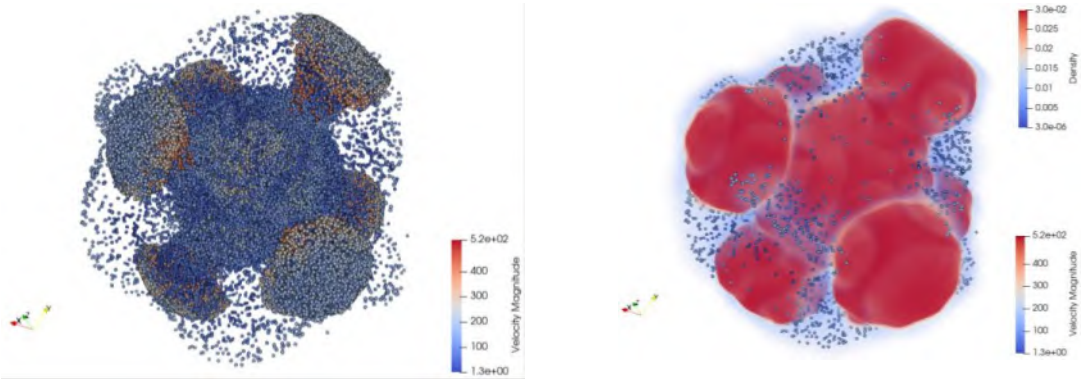


Рис. 4: Результат работы фрагментированной реализации PIC метода (на одном из временны́х шагов): распределение частиц (*а*); распределение частиц и плотности (*б*)

ментов (для фрагментированной реализации), 10^3 шагов по времени, частицы изначально распределены в облаке в центре расчетной области. Во втором варианте (рис. 6) использован фиксиро-

а *б*

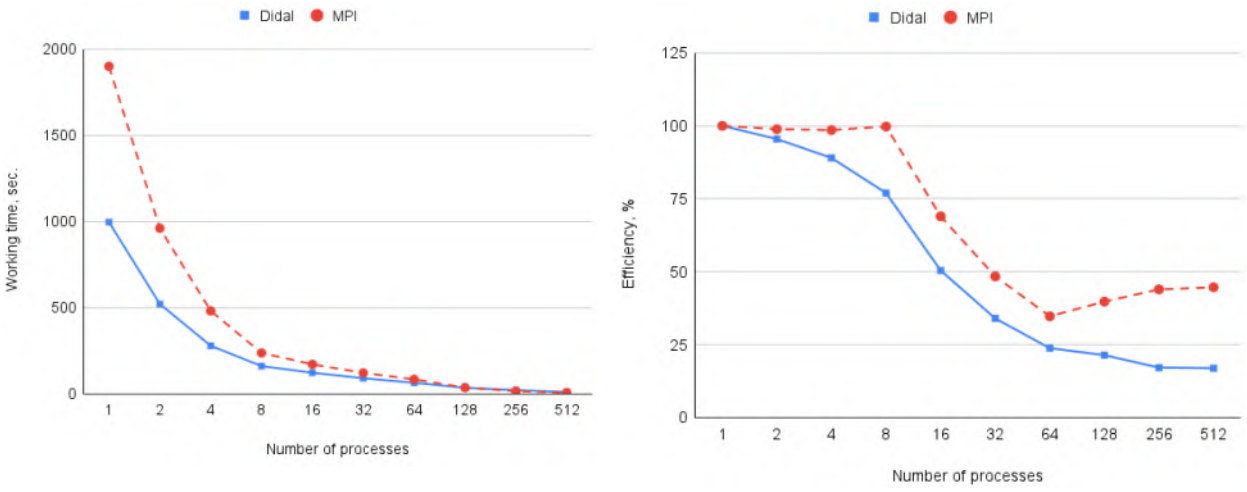


Рис. 5: Фиксированный размер задачи, время работы (*а*) и эффективность (*б*)

а *б*

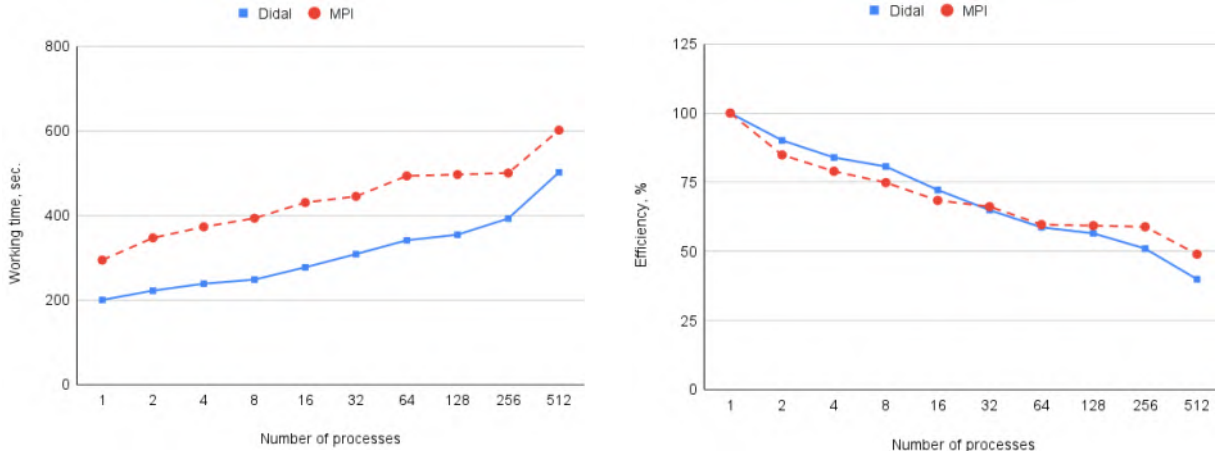


Рис. 6: Фиксированный размер задачи в каждом процессе, время работы (*а*) и эффективность (*б*)

ванный размер задачи на процесс (так называемая слабая масштабируемость) со следующими параметрами в каждом процессе: сетка 128^3 ячеек, 10^6 частиц, 8^3 фрагментов (для фрагментированной реализации), 10^3 шагов по времени, частицы изначально распределены равномерно в расчетной области.

Полученные результаты показывают, что фрагментированная реализация имеет лучшую производительность, чем аналогичная MPI-реализация, и сравнимую масштабируемость. Связано это как с улучшением работы с кэшем вследствие фрагментации, так и с тем, что Didal автоматически поддерживает асинхронные коммуникации, что позволяет обеспечить наложение коммуникаций и вычислений.

Полученные результаты показывают, что библиотека Didal может использоваться для создания эффективных фрагментированных реализаций сложных численных задач. В дальнейших планах – развитие библиотеки (в том числе реализация динамической балансировки нагрузки как универсальных алгоритмов, так и конкретно для PIC метода) и реализация с ее помощью фрагментированных версий других численных задач.

4. Разработан метод кодирования нуклеотидных последовательностей для использования библиотеки cuSTAR, реализующей ассоциативные вычисления на графических ускорителях, для решения задач биоинформатики. Проведен анализ способов в зависимости от размера занимаемой памяти и времени поиска отдельного нуклеотида в последовательности.

За последние несколько лет обработка генома стала широко востребованной задачей. Различными вариантами обработки занимаются как частные лаборатории (от выполнения ДНК-тестов до создания генетических паспортов), так и научные коллективы. При этом и первые, и вторые обрабатывают большие объемы данных как за счет количества образцов, так и за счет длины этих образцов: от десятков тысяч до нескольких миллиардов нуклеотидов. Кроме того, расширяется область применения молекулярно-генетических исследований (биомедицина, фармакология, нанобиоинженерия и т. д.). Заметим, что огромная часть вычислений связана с поиском отдельных нуклеотидов или их последовательностей как в большой последовательности, так и в большом числе последовательностей.

Для таких вычислений целесообразно использовать ассоциативные параллельные архитектуры, поскольку они специализируются на быстром поиске. Однако такие системы слабо представлены на рынке компьютерной техники в отличие от широкодоступных графических ускорителей. Библиотека cuSTAR была разработана для реализации абстрактной модели ассоциативных вычислений (STAR-машины) на графических ускорителях.

Данная работа представляет возможные способы организации данных биоинформатики для обработки системой cuSTAR. Это первая работа из предполагаемого цикла разработки ассоциативных алгоритмов для решения задач биоинформатики.

Последовательности нуклеотидов повсеместно представляются массивом символьного типа. Однако такое представление невозможно использовать в системе cuSTAR, поскольку типы данных для ассоциативной обработки в этой системе – это бинарные таблицы, столбцы и строки. Поэтому символы "A", "C", "G", "T", а также символ "–" для обозначения неизвестного или произвольного нуклеотида кодируются бинарными словами.

Первое представление использует только 3 бита, поэтому оно названо "mem_opt" (оптимальное по занимаемой памяти). Второе представление, time_opt, использует 4 бита: "1000" для аденина,

Таблица 2: Оценка размера памяти, необходимой для хранения нуклеотидных последовательностей разной длины

Метод	CPU (B)	GPU (KB)					
		100	1000	10000	100000	1000000	10000000
Mem_opt	72	0.05	0.38	3.68	36.63	366.21	3662.11
Time_opt	88	0.06	0.5	4.91	48.84	488.28	4882.81
char[]	8	0.1	0.98	9.77	97.66	976.56	9793.63

"0100" для цитазина, "0010" для гуанина и "0001" для тинина. С одной стороны, при таком представлении объем памяти увеличивается на 22 % на CPU и на 33 % на GPU (табл. 2). С другой стороны, каждому нуклеотиду соответствует определенный столбец таблицы, позиции символа "–" легко определяются. Для определения всех позиций заданного нуклеотида в последовательности вместо поиска слова в таблице требуется определить номер столбца, который этому нуклеотиду соответствует, что существенно сокращает время выполнения операции (рис. 7). При этом стоит отметить, что time_opt занимает в 2 раза меньше памяти на GPU, чем при стандартном представлении последовательности нуклеотидов в виде символьного массива.

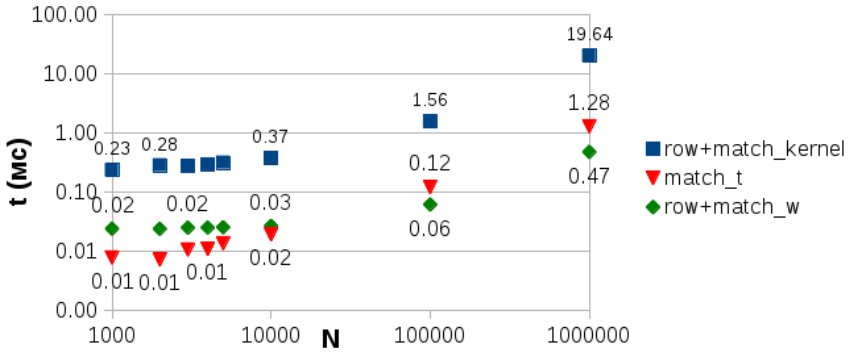


Рис. 7: Время поиска нуклеотида в последовательности для mem_opt (row+match_kernel) и для time_opt(match_t, row+match_w)

Таким образом, предложенные способы кодирования данных компактнее стандартного представления в виде массива символов. Представление time_opt позволяет проводить поиск нуклеотидов в последовательности на порядок быстрее, чем процедура из библиотеки стандартных ассоциативных алгоритмов. В то же время представление mem_opt предпочтительней в случае, если используется представление нуклеотидной последовательности в виде графовой структуры, например графа де Брёйна.

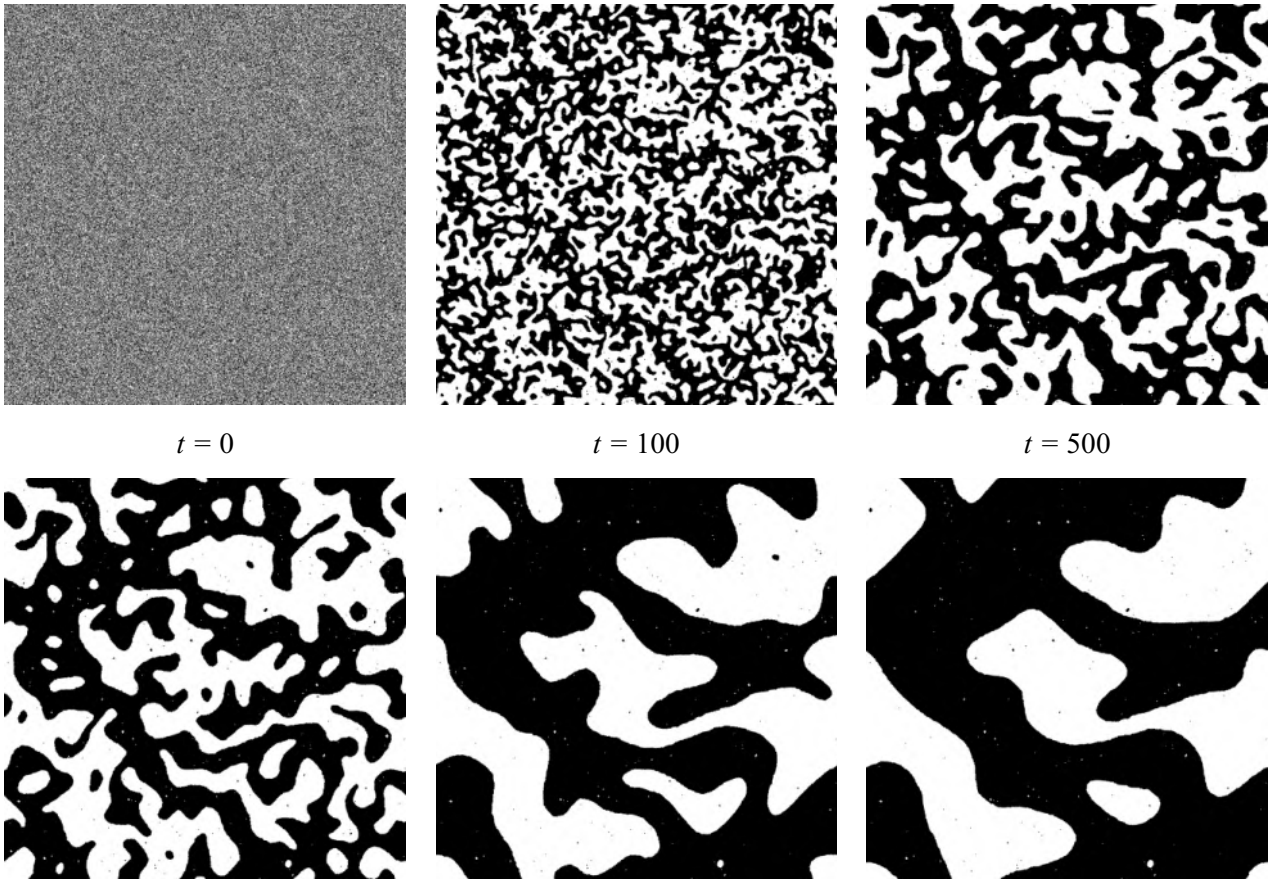
5. Созданы программные модули, реализующие имитационную модель разделения двух жидких фаз на поверхности вещества, с использованием библиотеки клеточно-автоматных топологий (под топологией понимается структура регулярного графа, вершинам которого соответствуют клетки массива, а ребрам – отношение соседства). Модули протестированы в последовательном режиме и на многопроцессорном вычислителе с общей памятью.

Разработаны программные модули – препроцессор, симулятор и постпроцессор, – реализующие дискретную имитационную модель разделения двух жидких фаз на поверхности вещества. Основой модели является двумерный синхронный клеточный автомат на квадратной решетке. Алфавит состояний клеток булев. Операционный режим клеточного автомата синхронный. При создании программных модулей применялась разрабатываемая в лаборатории на протяжении нескольких последних лет библиотека клеточно-автоматных топологий, предназначенная для

удобной и технологичной программной реализации клеточных автоматов как в последовательном варианте, так и на параллельных вычислительных системах.

Библиотека предоставляет разработанным модулям довольно широкий круг возможностей. В симуляторе функционал библиотеки используется для обеспечения обмена данными между клетками автомата на каждой итерации в соответствии с топологией клеточного массива, выбранной из расширяемого списка. Инициализация клеток массива при подготовке исходных данных препроцессором также реализуется библиотечными функциями. Постпроцессору библиотека обеспечивает извлечение состояний клеток для получения результата моделирования. Сохранение клеточного массива в файл, как и чтение из файла сохраненного клеточного массива во всех трех модулях, также производится путем обращения к библиотеке. Прикладному программисту остается только реализовать функции переходов клетки.

Работоспособность разработанных модулей и подключенных к ним библиотечных функций протестирована в последовательном режиме и на многопроцессорном вычислителе с общей памятью. С помощью новых модулей проведено компьютерное моделирование процесса коагуляции в эмульсии. На рис. 8 приведены результаты этого моделирования. Клеточный массив имеет размер 1000×1000 клеток, состояние каждой из которых показывает, какая из двух несмешивающихся фаз занимает эту клетку. В начальный момент времени ($t = 0$) фазы, окрашенные на рисунке черным и белым цветами, соответственно распределены по клеткам в случайном порядке равновероятно. В ходе моделирования ($t = 100,\ldots,10000$) наблюдается процесс коагуляции фаз,



$t = 0$ $t = 100$ $t = 500$

$t = 1000$ $t = 5000$ $t = 10000$

Рис. 8: Результаты компьютерного моделирования разделения
двух жидких фаз на поверхности вещества

составляющих эмульсию. Полученный результат подтвердил, что использование библиотеки клеточно-автоматных топологий в имитационном моделировании повышает качество программного кода и существенно сокращает трудозатраты квалифицированных программистов, требуемые для создания программной реализации клеточного автомата.

6. Разработана методика представления ряда свойств целенаправленно развивающихся систем в виде геометрических объектов при их моделировании (продолжающаяся разработка).

За отчетный период в рамках продолжения прошлогодних работ удалось представить некоторые новые свойства развивающихся систем в виде геометрических объектов. В сочетании с представлением траекторий развития как кривых в пространстве факторов (достижение 2021 г.) удалось представить процессы развития для двух стратегий разработки программных систем – последовательное и итеративное. В результате определены требования к технологичному инструментарию менеджера проекта, а также подготовлены спецификации этого инструментария как задание на разработку программной системы поддержки. Нашла свое подтверждение выдвинутая в прошлом году гипотеза о том, что геометрическое представление может быть использовано для общей модели системы автономных фрагментов, реализующих наиболее подходящие вычислительные системы.

В качестве иллюстративных примеров графического герметичного представления развивающихся систем на рис. 9–11 показаны схемы выполнения проектов из области разработки программных систем.

Главное преимущество развиваемого подхода относится к автоматизации деятельности менеджера проектов в части планирования и контроля выполнения планов, что достигается за счет

Рис. 9: Детализированное представление развития программного проекта

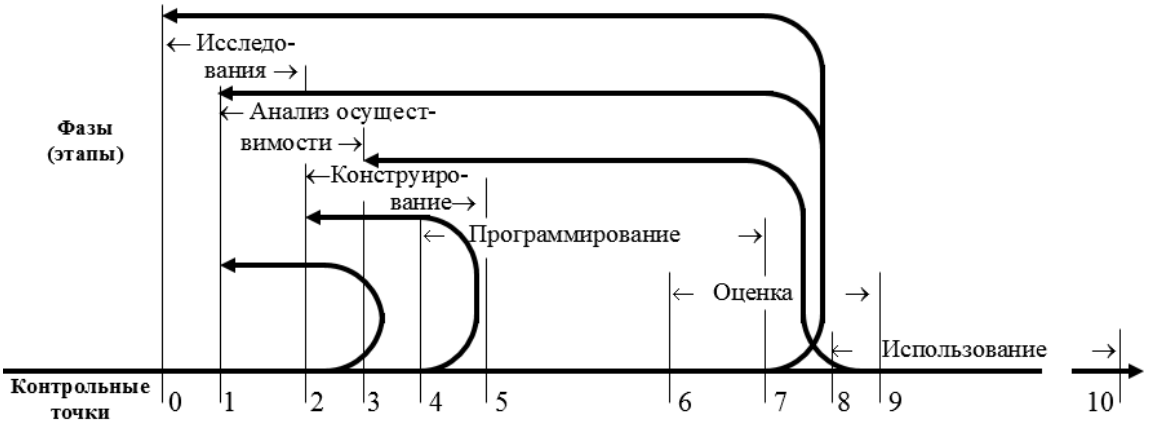


Рис. 10: Учет итеративности в модели фазы – функции Гантера

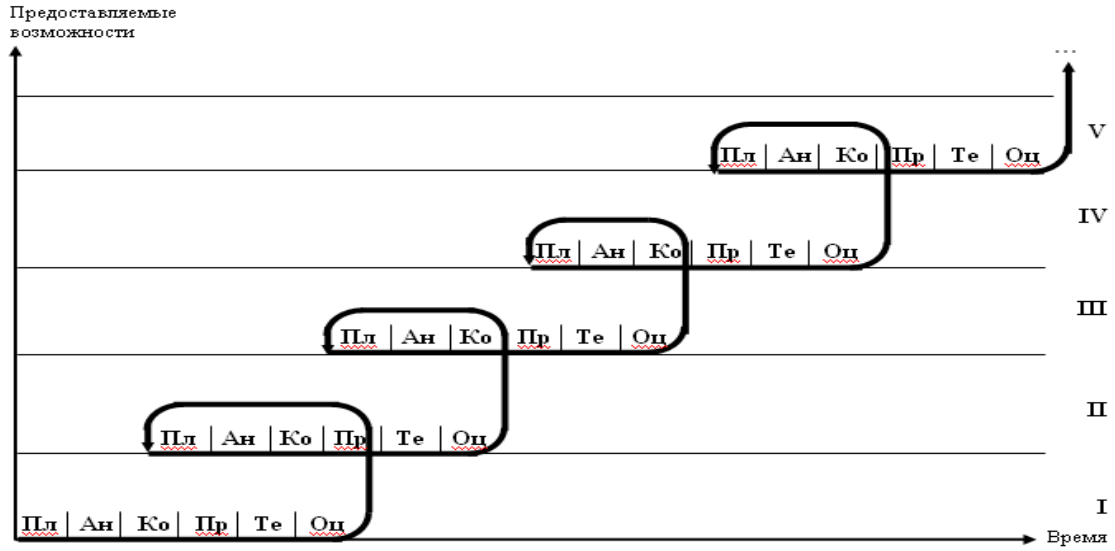


Рис. 11: Пять итераций цикла развития проекта в спиральной модели Г. Буча. Обозначения этапов, повторяющихся на каждой итерации: Пл – планирование, Ан – анализ, Ко – конструирование, Пр – программирование, Те – тестирование, Оц – оценка

задания семантической нагрузки и требуемых действий для элементов геометрического представления развития проекта. Дополнительные преимущества связаны с приписыванием элементам геометрического представления различных содержательных функций, поддерживающих деятельность менеджера.

Публикации

Издания, включенные в реферативную базу данных Scopus

1. Malyshkin, V., Vlasenko, A., Michurov, M. Automated debugging of fragmented programs in LuNA system / D. Balandin, K. Barkalov, I. Meyerov (eds.) Mathematical Modeling and Supercomputer Technologies. MMST 2022. Communications in Computer and Information Science. Vol. 1750. Springer, Cham. https://doi.org/10.1007/978-3-031-24145-1_22.

2. Malyshkin, V. Parallel computing technologies. Preface // J. Supercomput. 2022. Vol. 78, iss. 4. P. 6054–6055. DOI: 10.1007/s11227-011-0615-3.

Издания, включенные в библиографическую базу данных РИНЦ

1. Беляев, Н. А. Автоматическое конструирование высокопроизводительных параллельных программ для задач разреженной линейной алгебры в системе LuNA // Пробл. информ. 2022. № 3. С. 46–60. DOI: 10.24412/2073-0667-2022-3-46-60.

2. Малышкин, В. Э., Перепёлкин, В. А., Чмиль, А. С. Разработка подсистемы автоматизированного применения алгоритмов динамической балансировки нагрузки для системы LuNA // Пробл. информ. 2022. № 4. DOI: 10.24412/2073-0667-2022-4-107-119.

3. Власенко, А. Ю., Мичуров, М. А., Мустафин, Д. Э. Автоматизация отладки и балансировки нагрузки во фрагментированных программах // Пробл. информ. 2022. № 3. С. 61–76. DOI: 10.24412/2073-0667-2022-3-61-76.

4. Медведев, Ю. Г. Имитационное моделирование прохождения ламинарного потока через локальное сужение в трубе // Пробл. информ. 2022. № 2. С. 44–52. DOI: 10.24412/2073-0667-2022-2-44-52.

5. Снытникова, Т. В. Библиотека реализации ассоциативных вычислений на графических ускорителях cuSTAR: представление данных для задач биоинформатики // Марчуковские научные чтения – 2022 : Тез. Междунар. конф., Новосибирск, 3–7 окт. 2022 г. С. 97. DOI: 10.24412/cl-35065-2022-1-01-32.

6. Скопин, И. Н. Концепция модельного времени развивающихся систем // Марчуковские научные чтения. 2022 : Тез. Междунар. конф., 3–7 окт. 2022 г. С. 96. DOI: 10.24412/cl-35065-2022-1-01-31.

7. Щукин, Г. А. Параллельная фрагментированная реализация метода частиц-в-ячейках (PIC) с помощью библиотеки управления распределенными данными Didal // Марчуковские научные чтения – 2022 : Тез. Междунар. конф., 3–7 окт. 2022 г. С. 100. DOI: 10.24412/cl-35065-2022-1-01-36.

Участие в конференциях и совещаниях

1. 39-я Всероссийская летняя молодежная школа-конференция по параллельному программированию, 4–15 июля 2022 г., Новосибирск – 2 пленарных доклада (Перепёлкин В. А., Медведев Ю. Г.).

2. 16-я Международная научная конференция "Параллельные вычислительные технологии 2022" (ПаВТ'2022), г. Дубна (Московская обл.), 29–31 марта 2022 г. – 1 доклад (Киреев С. Е.).

3. Национальный Суперкомпьютерный Форум (НСКФ-2022), Переславль-Залесский, 29 ноября – 2 декабря 2022 г. – 1 доклад (Скопин И. Н.).

4. 5-я Международный образовательный форум "Алтай – Азия 2022: Евразийское образовательное пространство – новые вызовы и лучшие практики", Барнаул, 15–17 сентября 2022 г. – 1 доклад (Арыков С. Б.).

5. The 3rd International Workshop on advanced information and computation technologies and systems, Irkutsk, December, 2022 – 1 доклад (Скопин И. Н.).

6. 20-я Открытая всероссийская конференция "Преподавание информационных технологий в Российской Федерации – 2022", Москва, 19–20 мая 2022 г. – 1 доклад (Скопин И. Н.).

7. Международная конференция "Марчуковские научные чтения – 2022", Новосибирск, 3–7 октября 2022 г. – 3 доклада (Скопин И. Н., Снытникова Т. В., Щукин Г. А.).

8. Международная конференция "Математическое моделирование и суперкомпьютерные технологии", Нижний Новгород, 14–17 ноября – 1 доклад (Власенко А. Ю., Малышкин В. Э.).

Участие в организации научных мероприятий

1. Малышкин В. Э.:

– член организационного комитета 39-й Всероссийской летней молодежной школы-конференции по параллельному программированию, Новосибирск, 4–15 июля 2022 г.,

– член организационного комитета 38-й Зимней школы ИВМиМГ СО РАН по параллельному программированию, Новосибирск, 31 января – 4 февраля 2022 г.;

2. Арыков С. Б. – член организационного комитета 39-й Всероссийской летней молодежной школы-конференции по параллельному программированию, Новосибирск, 4–15 июля 2022 г.;

3. Власенко А. Ю.:

– член организационного комитета 39-й Всероссийской летней молодежной школы-конференции по параллельному программированию, Новосибирск, 4–15 июля 2022 г.,

– член организационного комитета 38-й Зимней школы ИВМиМГ СО РАН по параллельному программированию, Новосибирск, 31 января – 4 февраля 2022 г.;

4. Городничев М.А.:

– член организационного комитета 39-й Всероссийской летней молодежной школы-конференции по параллельному программированию, Новосибирск, 4–15 июля 2022 г.,

– член организационного комитета 38-й Зимней школы ИВМиМГ СО РАН по параллельному программированию, Новосибирск, 31 января – 4 февраля 2022 г.;

– член организационного комитета конференции "Установки мегасайнс и медиумсайнс: вопросы диверсификации программно-аппаратного обеспечения и информационной безопасности", Новосибирск, 12–13 декабря 2022 г.

5. Киреев С. Е.:

– член организационного комитета 39-й Всероссийской летней молодежной школы-конференции по параллельному программированию, Новосибирск, 4–15 июля 2022 г.,

– член организационного комитета 38-й Зимней школы ИВМиМГ СО РАН по параллельному программированию, Новосибирск, 31 января – 4 февраля 2022 г.;

6. Калгин К. В. – член организационного комитета 39-й Всероссийской летней молодежной школы-конференции по параллельному программированию, Новосибирск, 4–15 июля 2022 г.,

7. Маркова В. П:

– член организационного комитета 39-й Всероссийской летней молодежной школы-конференции по параллельному программированию, Новосибирск, 4–15 июля 2022 г.,

– член организационного комитета 38-й Зимней школы ИВМиМГ СО РАН по параллельному программированию, Новосибирск, 31 января – 4 февраля 2022 г.;

8. Медведев Ю. Г:

– член организационного комитета 14-й Международной конференции "Новые информационные технологии в исследовании сложных структур" (ICAM-2022), оз. Байкал, 19–24 сентября 2022 г.,

– член программного комитета 14-й Международной конференции "Новые информационные технологии в исследовании сложных структур" (ICAM-2022), оз. Байкал, 19–24 сентября 2022 г.,

9. Перепёлкин В. А.:

– член организационного комитета 39-й Всероссийской летней молодежной школы-конференции по параллельному программированию, Новосибирск, 4–15 июля 2022 г.,

– член организационного комитета 38-й Зимней школы ИВМиМГ СО РАН по параллельному программированию, Новосибирск, 31 января – 4 февраля 2022 г.;

10. Щукин Г.А.:

– член организационного комитета 39-й Всероссийской летней молодежной школы-конференции по параллельному программированию, Новосибирск, 4–15 июля 2022 г.,

– член организационного комитета 38-й Зимней школы ИВМиМГ СО РАН по параллельному программированию, Новосибирск, 31 января – 4 февраля 2022 г.

Итоговые данные по лаборатории

Публикаций, индексируемых в базе данных Scopus – 2
Публикаций, индексируемых в базе данных РИНЦ – 9
Докладов на конференциях – 11, в том числе 2 пленарных
Участников оргкомитетов конференций – 19

Кадровый состав

1. Малышкин В. Э.	г.н.с., зав. лабораторией	д.т.н.
2. Арыков С. Б.	м.н.с.	к.ф.-м.н.
3. Власенко А. Ю.	н.с.	к.т.н.
4. Городничев М. А.	н.с.	
5. Калгин К. В.	м.н.с.	к.ф.-м.н.
6. Киреев С. Е.	н.с.	
7. Маркова В. П.	с.н.с.	к.т.н.
8. Медведев Ю. Г.	с.н.с.	к.т.н.
9. Перепёлкин В. А.	н.с.	
10. Савукова В. А.	ст. техник	
11. Скопин И. Н.	с.н.с.	к.ф.-м.н.
12. Снытникова Т. В.	м.н.с.	к.т.н.
13. Щукин Г. А.	н.с.	

Педагогическая деятельность

Малышкин В. Э.	– профессор, зав. кафедрой, НГУ; профессор, зав. кафедрой НГТУ
Арыков С. Б.	– доцент НГТУ
Власенко А. Ю.	– доцент НГУ
Маркова В. П.	– доцент НГУ, НГТУ
Медведев Ю. Г.	– доцент НГУ
Городничев М. А.	– ст. преподаватель НГУ, ассистент НГТУ
Калгин К. В.	– ст. преподаватель НГУ
Киреев С. Е.	– ст. преподаватель НГУ

Перепёлкин В. А. – ст. преподаватель НГУ, НГТУ
Щукин Г. А. – ассистент НГТУ

Руководство студентами

Абрамушкина Е. С. – 4-й курс ФИТ НГУ, руководитель Киреев С. Е.
Алексейчук С. Д. – 3-й курс ФПМИ НГТУ, руководитель Арыков С. Б.
Балашов А. А. – 3-й курс ФПМИ НГТУ, руководитель Арыков С. Б.
Баранов И. Н. – 1-й курс магистратуры ФИТ НГУ, руководитель Городничев М. А.
Белявцев Б. В. – 3-й курс ФИТ НГУ, руководитель Перепёлкин В. А.
Богданович П. Ю. – 3-й курс ФПМИ НГТУ, руководитель Арыков С. Б.
Болдырев С. Д. – 3-й курс ФПМИ НГТУ, руководитель Арыков С. Б.
Бондарев В. Д. – 3-й курс ФИТ НГУ, руководитель Медведев Ю. Г.
Брусникин М. А. – 4-й курс ФИТ НГУ, руководитель Перепёлкин В. А.
Бурнышев Е. К. – 3-й курс ФИТ НГУ, руководитель Медведев Ю. Г.
Валитов А. А. – 4-й курс ФИТ НГУ, руководитель Перепёлкин В. А.
Васильев И. А. – 3-й курс ММФ НГУ, руководитель Скопин И. Н.
Галишева К. А. – 4-й курс ФИТ НГУ, руководитель Перепёлкин В. А.
Гемуев А. Б. – 3-й курс ФИТ НГУ, руководитель Перепёлкин В. А.
Глебов Т. В. – 3-й курс ФПМИ НГТУ, руководитель Арыков С. Б.
Голиков М. О. – 2-й курс магистратуры ФИТ НГУ, руководитель Киреев С. Е.
Дастай-оол С. А. – 3-й курс, 4-й курс ФПМИ НГТУ, руководитель Городничев М. А.
Дорн А. А. – 4-й курс ФИТ НГУ, руководитель Перепёлкин В. А.
Дягилев М. А. – 3-й курс ФИТ НГУ, руководитель Городничев М. А.
Еделев А. О. – 3-й курс ФПМИ НГТУ, руководитель Арыков С. Б.
Ефимцев Ф. А. – 1-й курс магистратуры ФИТ НГУ, руководитель Городничев М. А.
Закиров В. К. – 2-й курс магистратуры ФИТ НГУ, руководитель Перепёлкин В. А.
Ивакин А. О. – 4-й курс ФИТ НГУ, руководитель Перепёлкин В. А.
Иванченко Д. В. – 4-й курс ФИТ НГУ, руководитель Перепёлкин В. А.
Капралова Р. Е. – 4-й курс ФИТ НГУ, руководитель Перепёлкин В. А.
Квашнин А. А. – 3-й курс ФИТ НГУ, руководитель Городничев М. А.
Кирсанов И. Д. – 3-й курс ММФ НГУ, руководитель Скопин И. Н.
Киселев К. С. – 3-й курс ФИТ НГУ, руководитель Киреев С. Е.
Клейменов Д. И. – 4-й курс ММФ НГУ, руководитель Скопин И. Н.
Колесников А. А. – 3-й курс ФПМИ НГТУ, руководитель Арыков С. Б.
Котенок В. И. – 3-й курс ММФ НГУ, руководитель Скопин И. Н.
Кразер В. Л. – 3-й курс ММФ НГУ, руководитель Скопин И. Н.
Крупская В. А. – 1-й курс магистратуры ФИТ НГУ, руководитель Городничев М. А.
Кудрявцев А. А. – 4-й курс ФИТ НГУ, руководитель Перепёлкин В. А.
Кузнецов К. В. – 3-й курс ФПМИ НГТУ, руководитель Арыков С. Б.
Курбатов М. А. – 3-й курс ФИТ НГУ, руководитель Власенко А. Ю.
Лебедев Н. В. – 4-й курс ФПМИ НГТУ, руководитель Арыков С. Б.
Левченко К. К. – 1-й курс магистратуры ФИТ НГУ, руководитель Перепёлкин В. А.
Леонтьева М. К. – 3-й курс ФИТ НГУ, руководитель Медведев Ю. Г.

Лямин А. С. – 2-й курс магистратуры ФИТ НГУ, руководитель Перепёлкин В. А.
Малханов В. В. – 2-й курс магистратуры ФИТ НГУ, руководитель Перепёлкин В. А.
Матус Н. Е. – 3-й курс ФИТ НГУ, руководитель Перепёлкин В. А.
Мичуров М. А. – 1-й курс магистратуры ФИТ НГУ, руководитель Власенко А. Ю.
Морозов Р. С. – 3-й курс ФИТ НГУ, руководитель Перепёлкин В. А.
Муратов М. А. – 3-й курс ФИТ НГУ, руководитель Перепёлкин В. А.
Мухиддинов Д. Х. – 2-й курс магистратуры ММФ НГУ, руководитель Скопин И. Н.
Нуштаев Ю. Ю. – 4-й курс ФИТ НГУ, руководитель Перепёлкин В. А.
Пирожков А. К. – 4-й курс ФИТ НГУ, руководитель Перепёлкин В. А.
Половникова П. С. – 4-й курс ФПМИ НГТУ, руководитель Арыков С. Б.
Попов Р. Д. – 4-й курс ФИТ НГУ, руководитель Медведев Ю. Г.
Пчелинцев С. Е. – 3-й курс ФИТ НГУ, руководитель Медведев Ю. Г.
Синюков В. К. – 3-й курс ФИТ НГУ, руководитель Перепёлкин В. А.
Скрипникова В. С. – 3-й курс ФИТ НГУ, руководитель Медведев Ю. Г.
Спирин В. А. – 4-й курс ФИТ НГУ, руководитель Перепёлкин В. А.
Толкачева Р. Э. – 3-й курс ФИТ НГУ, руководитель Городничев М. А.
Трубицына Ю. С. – 4-й курс ФИТ НГУ, руководитель Медведев Ю. Г.
Трынкин М. А. – 3-й курс ФИТ НГУ, руководитель Городничев М. А.
Уточкин С. Е. – 4-й курс ФИТ НГУ, руководитель Городничев М. А.
Федоров В. Е. – 3-й курс ФПМИ НГТУ, руководитель Арыков С. Б.
Хайруллаев У. Б. – 2-й курс магистратуры ММФ НГУ, руководитель Скопин И. Н.
Царев В. Д. – 3-й курс ФИТ НГУ, руководитель Власенко А. Ю.
Чемагин А. С. – 1-й курс магистратуры ФИТ НГУ, руководитель Киреев С. Е.
Шевченко С. В. – 4-й курс ММФ НГУ, руководитель Скопин И. Н.

Руководство аспирантами

Артюхов А. А. – 2-й курс аспирантуры ФИТ НГУ, руководитель Малышкин В. Э.
Кудинов А. Ю. – 1-й курс аспирантуры ИВМиМГ СО РАН, руководитель Малышкин В. Э.

Защита дипломов

Баранов И. Н. – бакалавр ФИТ НГУ, руководитель Киреев С. Е.
Ефимцев Ф. А. – бакалавр ФПМИ НГТУ, руководитель Городничев М. А.
Кошкарев А. В. – бакалавр ФПМИ НГТУ, руководитель Арыков С. Б.
Крупская В. А. – бакалавр ФПМИ НГТУ, руководитель Городничев М. А.
Кудинов А. Ю. – магистр ФИТ НГУ, руководитель Городничев М. А.
Левченко К. К. – бакалавр ФИТ НГУ, руководитель Перепёлкин В. А.
Мичуров М. А. – бакалавр ФИТ НГУ, руководитель Власенко А. Ю.
Мустафин Д. Э. – бакалавр ФИТ НГУ, руководитель Власенко А. Ю.
Налепова Е. Д. – бакалавр ФИТ НГУ, руководитель Городничев М. А.
Чемагин А. С. – бакалавр ФИТ НГУ, руководитель Киреев С. Е.
Чмиль А. В. – магистр ФИТ НГУ, руководитель Перепёлкин В. А.