МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИМЕНИ Н. Г. ЧЕРНЫШЕВСКОГО»

Кафедра дискретной математики и информационных технологий

СРАВНИТЕЛЬНОЕ ИССЛЕДОВАНИЕ АЛГОРИТМОВ ПРОСТРАНСТВЕННОЙ ОПТИМИЗАЦИИ НА ПРАКТИЧЕСКОЙ РЕАЛИЗАЦИИ ЗАДАЧИ РАССАДКИ

АВТОРЕФЕРАТ БАКАЛАВРСКОЙ РАБОТЫ

студента 4 курса 421 группы
направления 09.03.01 — Информатика и вычислительная техника
факультета КНиИТ
Аширова Егора Игоревича

Научный руководитель
старший преподаватель _____ П. О. Дмитриев

Заведующий кафедрой
доцент, к. ф.-м. н. _____ Л. Б. Тяпаев

Саратов 2026

ВВЕДЕНИЕ

Организация крупных олимпиад и письменных контрольных мероприятий неизбежно сталкивается с практической проблемой: как распределить участников по местам так, чтобы минимизировать возможность несанкционированного взаимодействия между знакомыми. Для школьных олимпиад это означает необходимость разводить на расстояние учеников из одного класса, одной школы или просто имеющих общие социальные связи. При ручной рассадке нескольких сотен участников задача становится чрезвычайно трудоёмкой и на практике решается приближённо: неизбежно остаются нарушения, а исправление каждого из них может порождать новые.

Помимо очевидной трудоёмкости, ручной подход требует отдельного оформления сопроводительных документов. При большом числе участников подготовка этих материалов занимает значительное время и сопряжена с риском ошибок, что обусловливает целесообразность автоматизации данного процесса.

Целью настоящей работы является разработка и сравнительное исследование алгоритмов пространственной оптимизации применительно к задаче рассадки участников олимпиады, а также создание программного обеспечения, автоматизирующего этот процесс.

Для достижения поставленной цели в работе решены следующие задачи:

1. проанализировать предметную область, обосновать актуальность задачи и выявить недостатки существующих подходов к рассадке участников массовых мероприятий;
2. формализовать задачу рассадки как задачу комбинаторной оптимизации и разработать систему штрафов, отражающую реальные ограничения на соседство участников;
3. обосновать выбор алгоритмов для сравнительного исследования и описать их теоретические основы;
4. реализовать выбранные алгоритмы оптимизации, адаптировав их к специфике задачи рассадки;
5. разработать программное обеспечение с графическим интерфейсом, провести сравнительное тестирование алгоритмов и реализовать экспорт результатов в форматы, пригодные для практического применения.

Объектом исследования является процесс автоматизированного распределения участников олимпиады по местам в аудиториях. Предметом исследования — алгоритмы комбинаторной оптимизации и методы их сравнения применительно к данному классу задач.

В работе применены методы стохастической оптимизации, эволюционных вычислений, а также сравнительный анализ по нескольким критериям качества. Реализация выполнена на языке Python 3.12 с применением библиотек PyQt6, pandas, matplotlib и numpy.

Практическая значимость работы состоит в том, что разработанное программное обеспечение может быть непосредственно применено при проведении олимпиад и иных мероприятий, требующих оптимизированного размещения участников по аудиториям.

Бакалаврская работа состоит из введения, пяти разделов («Анализ предметной области», «Постановка и формализация задачи», «Алгоритмы пространственной оптимизации», «Программная реализация», «Тестирование и сравнение алгоритмов»), заключения, списка использованных источников и четырёх приложений.

КРАТКОЕ СОДЕРЖАНИЕ РАБОТЫ

В первом разделе «Анализ предметной области» рассматривается актуальность задачи рассадки участников олимпиад. При нескольких сотнях человек из разных классов, школ и муниципалитетов ручной перебор занимает значительное время, а исправление одной конфликтной пары может порождать новые нарушения. Программа дополнительно автоматизирует оформление сопроводительных документов: по завершении рассадки формируются файлы для вахты, дежурных в аудиториях и учителей.

Проведён анализ двух категорий существующих решений. Системы управления мероприятиями позволяют создавать схемы залов и назначать участников на места, однако не располагают механизмами ограничений на соседство конкретных пар. Платформы для проведения олимпиад реализуют рассадку в виде алфавитного или случайного распределения без оптимизационных критериев. Задача автоматизированной рассадки с минимизацией штрафов за соседство знакомых остаётся практически не закрытой существующими инструментами. В математическом смысле она является частным случаем задачи квадратичного назначения (QAP) [1].

Во втором разделе «Постановка и формализация задачи» задача рассадки формализована как задача комбинаторной оптимизации.

Пусть задано конечное множество участников $S = \{s_1, s_2, \ldots, s_n\}$. Каждый участник s_i характеризуется атрибутами: ФИО, образовательная организация, класс (от 1 до 11), буква класса, город, муниципальное образование и необязательная метка знакомства a_i. Конфигурация залов задаётся набором аудиторий $\mathcal{C} = \{c_1, c_2, \ldots, c_m\}$; каждая аудитория c_k описывается двоичной матрицей $M_k \in \{0,1\}^{R_k \times T_k}$, где $M_k[r][t] = 1$ означает наличие физического места. Пустая клетка разрывает соседство, что позволяет моделировать залы произвольной планировки. Рассадкой называется инъективное отображение $X\colon S \to \bigcup_{k=1}^{m} \mathrm{seats}(c_k)$.

Разработана система штрафов за соседство, представленная в таблице 1. Критерий явного знакомства имеет приоритет: если оба участника имеют одинаковую непустую метку $a_i = a_j$, штраф равен 70 вне зависимости от школы и города.

Таблица 1 – Базовые штрафы за соседство участников

Ситуация	**Базовый штраф**
Совпадает метка знакомства $a_i = a_j \neq \emptyset$	70
Разные школы или разные города	0
Одна школа, разные классы	10
Один класс, разные буквы	30
Одноклассники (совпадают класс и буква)	70

Вводится коэффициент типа соседства: $k(\tau) = 1{,}0$ для прямого соседства ($side$) и $k(\tau) = 0{,}5$ для диагонального ($diag$), поскольку диагональные соседи физически дальше. Штраф за конкретную пару соседей: $p(s_i, s_j, \tau) = b(s_i, s_j) \cdot k(\tau)$; целевая функция — минимизация суммарного штрафа $P(X)$.

Поскольку точный максимум штрафов найти так же NP-трудно, как и минимум, для нормировки результатов вычисляется приближённая верхняя граница $P_{\max}$ с помощью вспомогательного алгоритма имитации отжига на максимизацию. Значение $P(X)/P_{\max} \cdot 100\%$ используется как сравнительный показатель качества рассадки.

Для сравнительного исследования алгоритмов выбраны четыре критерия: качество решения (нормированный штраф), время работы, стабильность (диапазон разброса [мин.; макс.] по серии из 10 запусков) и скорость сходимости.

В третьем разделе «Алгоритмы пространственной оптимизации» обосновывается выбор трёх алгоритмов. Задача рассадки является NP-трудной [2], сводясь к задаче квадратичного назначения (QAP). Три конкретных алгоритма выбраны так, чтобы охватить принципиально разные парадигмы оптимизации: детерминированные эвристики, стохастический поиск и эволюционные вычисления.

Жадный алгоритм с мультистартом и итерированным локальным поиском [3] состоит из четырёх фаз. На первой выполняется мультистарт из 10 независимых жадных рассадок: участники группируются по ключу (город, школа, класс, буква) и сортируются по убыванию размера группы; для каждого места выбирается участник с нулевым или минимальным штрафом; наилучшая из 10 рассадок передаётся в следующие фазы. Вторая и третья фазы — итерированный локальный поиск [4]: поочерёдно выполняются оптимизация перемещениями (перевод участника на свободное место)

и оптимизация свапами (обмен двух участников местами). Четвёртая фаза повторяет цикл до тех пор, пока он приносит улучшение. Дельта штрафа вычисляется инкрементально — пересчитываются только вклады изменившихся мест.

Алгоритм имитации отжига [5]. Вероятность принятия ухудшающего хода $\Delta > 0$: $P(\text{принять}) = \exp(-\Delta/T)$. Параметры вычисляются адаптивно: $I_{\max} = \max(20\,000;\ n \cdot 200)$, $T_0 = \max(1{,}0;\ P_0 \cdot 0{,}1)$, температура охлаждается геометрически до $T_{\text{fin}} = 1{,}0$. Ключевым улучшением является механизм направленного выбора нарушителей: поддерживается множество V мест с ненулевым штрафом; вероятность выбора нарушителя $p_{\text{guided}} = (T_0 - T)/(T_0 - T_{\text{fin}})$ нарастает линейно по мере охлаждения — от широкого исследования в начале до целенаправленной локальной оптимизации в конце. Алгоритм хранит лучшее найденное решение за всё время работы.

Генетический алгоритм [6]. Особь кодируется списком из n индексов мест; допустимость решения гарантируется структурой представления. Начальная популяция из 50 особей формируется случайными выборками. Родители выбираются турнирной селекцией с размером турнира 3. Применяется оператор OX (Order Crossover): из первого родителя берётся случайный подотрезок и копируется в потомка; оставшиеся позиции заполняются из второго родителя в порядке их встречаемости. Вероятность мутации 0,02 адаптивно усиливается при стагнации. После скрещивания к потомку применяется локальный поиск; два лучших индивида переходят в следующее поколение без изменений.

Все три алгоритма используют единую функцию штрафов, что обеспечивает объективное сравнение результатов в едином метрическом пространстве.

В четвёртом разделе «Программная реализация» описывается разработанное настольное приложение для операционной системы Windows, реализованное на языке Python 3.12 с использованием библиотеки PyQt6. Приложение организовано по модульному принципу и разделено на пять пакетов: `models` — модели данных; `core` — ядро (функция штрафов, загрузчик, валидатор); `algorithms` — три алгоритма и SA-максимизатор; `export` — Excel-файлы и HTML-отчёт; `ui` — графический интерфейс. Главное окно при запуске программы показано на рисунке 1.

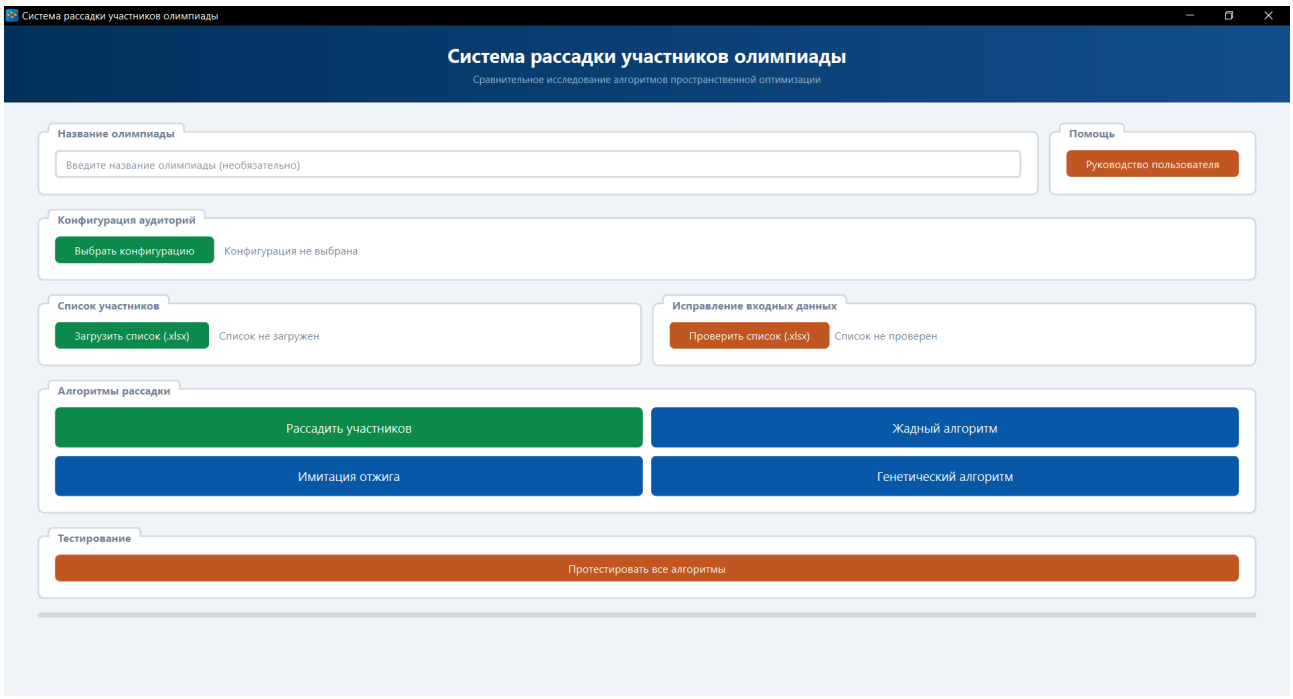

Рисунок 1 – Главное окно при запуске программы

Аудитория описывается двоичной матрицей размером до 20×20. Редактор матрицы (рисунок 2) поддерживает рисование прямоугольников мышью, масштабирование, панорамирование и историю undo/redo до 50 снимков состояния. Метка места формируется автоматически: буква — порядковый номер занятого столбца, цифра — номер места в столбце сверху вниз. Пустые клетки разрывают соседство, обеспечивая корректную работу с залами произвольной планировки.

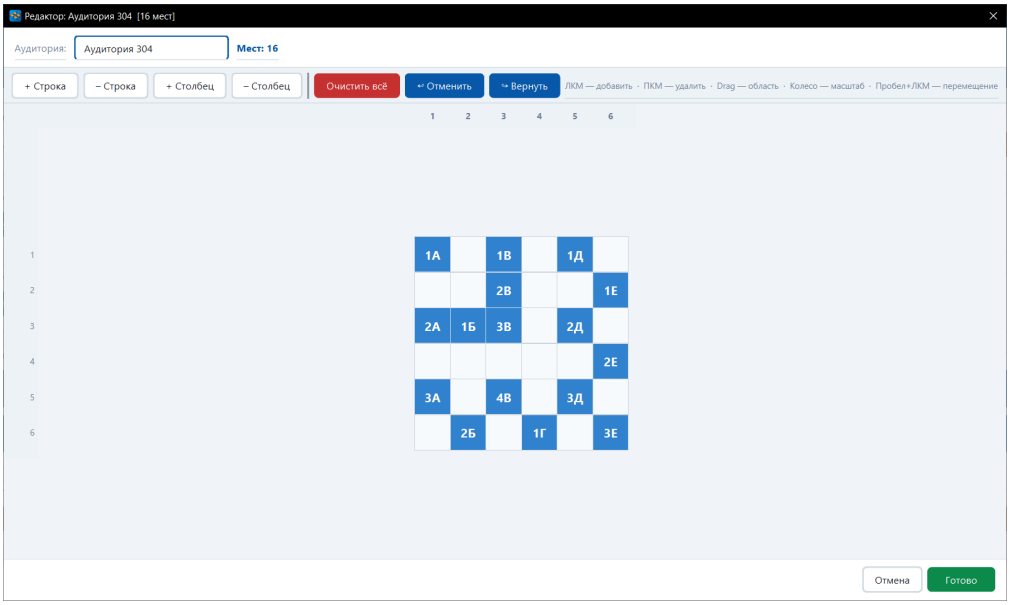

Рисунок 2 – Редактор матрицы аудитории

Входные данные загружаются из файла xlsx. Список участников проверяется пятиступенчатым валидатором [7]: пустые обязательные поля, полные дубликаты, записи с одинаковым ФИО но разными данными, похожие имена и названия школ. Найденные проблемы отображаются в диалоге-мастере, где пользователь может исправить их до запуска алгоритмов. Рабочий процесс построен линейно: выбор конфигурации, загрузка и проверка списка, запуск алгоритма. По завершении рассадки отображается диалог с суммарным штрафом, его долей от $P_{\max}$ и временем работы.

После завершения рассадки автоматически формируется несколько выходных Excel-файлов: «Вахта.xlsx» для сотрудника вахты, отдельный файл по каждой аудитории с указанием конкретного места и файл «Для учителя.xlsx» со всеми исходными столбцами и добавленными столбцами «Место» и «Аудитория».

Режим сравнительного тестирования запускает каждый из трёх алгоритмов 10 раз на одних и тех же данных. По завершении генерируется самодостаточный HTML-отчёт со сводной таблицей статистики и визуализацией: графики стабильности, сходимости и состава штрафов по шести категориям нарушений.

В пятом разделе «Тестирование и сравнение алгоритмов» качество рассадки выражается в долях от опорного максимума $P_{\max}$; нулевой штраф означает полностью бесконфликтную рассадку. В сводных таблицах приняты обозначения: «Лучший» — минимальный штраф за серию, «Ср.» — средней штраф по серии, «Макс.» — максимальный штраф за серию, «Ср. t» — среднее время выполнения одного запуска.

Базовый сценарий. Пять стандартных аудиторий, по 25 мест в сетке 5×5, итого 125 мест; 125 участников из 34 школ (классы 6 и 7), $P_{\max} = 8275{,}0$. Все три алгоритма дают нулевой штраф в 100% из 100 запусков. Жадный алгоритм справляется за 0,05 с, что в 12 и 74 раза быстрее имитации отжига и генетического алгоритма соответственно.

Большой класс: нижняя граница штрафа. Те же 125 мест; одна школа выставила 58 участников одного класса. При такой концентрации одноклассников ненулевой штраф математически неизбежен: единственный способ разместить одноклассников без штрафа — занять места с зазором по обоим осям, что даёт не более 9 из 25 мест в аудитории. Опорный максимум

$P_{\max} = 8810{,}0$. Все три алгоритма сходятся к одному и тому же минимуму — штрафу 240,0 (2,7%), что является нижней границей задачи. Жадный алгоритм абсолютно стабилен (нулевое отклонение), тогда как стохастические методы иногда незначительно отклоняются от оптимума.

Несколько возрастных групп. Те же 125 мест, но участники из 34 школ распределены по классам 6–9. Разнородность возрастов существенно облегчает задачу: разные классы одной школы дают штраф лишь 10, а разные школы — 0, что снижает вероятность конфликтного соседства. Опорный максимум $P_{\max} = 5690{,}0$. Жадный алгоритм и имитация отжига достигают нулевого штрафа в 100% запусков. Генетический алгоритм иногда нестабилен, но его лучший результат также равен нулю. Задача с неоднородным набором классов решается значительно проще, чем базовый сценарий.

Слабая вариация параметров. При 100% заполненности конкурирующие ограничения по четырём крупным школам затрудняют одновременную оптимизацию. Жадный алгоритм показывает лучший результат в серии (0%) и работает в 120 раз быстрее генетического. Имитация отжига и генетический алгоритм не достигают нулевого штрафа ни в одном из 10 запусков (0,1% и 0,3% соответственно).

Нестандартные планировки. Аудитория 303 содержит 50 мест, организованных в два блока по 5 столбцов с центральным проходом. Аудитория 207 содержит 25 мест в матрице 7×7 с многочисленными пропусками — L-образный контур с несколькими изолированными блоками. Все три алгоритма достигают нулевого штрафа в 100% запусков; время работы даже ниже базового, поскольку разрывы в матрице уменьшают число соседних пар.

Масштабируемость: 2000 участников. Восемьдесят аудиторий 5×5, итого 2000 мест; 2000 участников из 120 школ, $P_{\max} = 75065{,}0$. Все три алгоритма дают нулевой штраф; адаптивное масштабирование параметров обеспечивает корректную работу без ручной настройки при любом числе участников. Жадный алгоритм справляется за 0,38 с, генетический — за 7,12 с.

Сравнение с ручной рассадкой (300 участников). Алгоритмическое решение (300 участников). 12 аудиторий по 25 мест, всего 300 участников из 33 школ (классы 7–9). Опорный максимум $P_{\max} = 16735{,}0$. Все три алгоритма дают нулевой штраф в 100% запусков. Жадный алгоритм справляется за 0,04 с.

Ручная рассадка и сравнение. Ручная рассадка тех же 300 участников потребовала последовательного распределения по аудиториям с проверкой всех восьми соседей. Каждое перемещение порождало каскадные исправления. Полный цикл, включая ручную проверку и оформление результатов занял 4–4,5 часа. Разработанная программа автоматически формирует выходные файлы. Жадный алгоритм выполнил ту же задачу за 0,04 с с нулевым штрафом.

По результатам тестирования **Жадный алгоритм** показывает наилучшее соотношение скорости и качества на однородных данных: при небольшом числе школ и стандартных аудиториях он стабильно находит нулевой штраф за 0,03–0,06 с. **Имитация отжига** является наиболее сбалансированным алгоритмом: критерий Метрополиса позволяет преодолевать локальные оптимумы, а направленный выбор нарушителей концентрирует поиск на проблемных парах; алгоритм обеспечивает стабильно хорошее качество при умеренном времени работы. **Генетический алгоритм** наиболее эффективен при максимально разнородных входных данных: множество разных школ, нестандартные планировки, смешанные возрастные группы; в таких условиях популяционный поиск с OX-скрещиванием объединяет частичные решения нескольких особей и обнаруживает структуры расстановки, недостижимые для одиночных методов.

ЗАКЛЮЧЕНИЕ

Цель настоящей работы — разработка и сравнительное исследование алгоритмов пространственной оптимизации применительно к задаче рассадки участников олимпиады — достигнута в полном объёме. Все пять поставленных задач решены.

В ходе анализа предметной области установлено, что существующие инструменты для организации мероприятий не позволяют автоматически размещать участников с учётом ограничений на соседство знакомых и произвольной планировки залов. Это подтвердило актуальность задачи и обосновало целесообразность разработки оригинального решения.

Задача рассадки формализована как задача комбинаторной оптимизации. Разработана система штрафов, отражающая степень знакомства участников и тип пространственного соседства. Единая реализация функции штрафов обеспечила сравнение алгоритмов в одном метрическом пространстве.

Обоснован выбор и выполнена реализация трёх алгоритмов, охватывающих принципиально разные парадигмы оптимизации: детерминированную эвристику, стохастический локальный поиск и эволюционные вычисления. Каждый алгоритм адаптирован к специфике задачи рассадки.

Разработано программное обеспечение с графическим интерфейсом, включающее редактор планировок аудиторий, валидатор входных данных и экспорт результатов. Проведено сравнительное тестирование на наборе сценариев с различными структурами входных данных и масштабами задачи; по результатам тестирования определены условия, при которых каждый из алгоритмов является предпочтительным.

Разработанное программное обеспечение готово к практическому применению для проведения олимпиад и иных письменных мероприятий, требующих оптимизированного размещения участников по аудиториям. Применение системы существенно снижает трудозатраты организаторов: полный цикл рассадки и формирование сопроводительных документов выполняются за секунды, тогда как ручное решение аналогичной задачи требует значительных временных затрат. Модульная архитектура и единая функция штрафов обеспечивают возможность дальнейшего развития системы и её адаптации к смежным сценариям без изменения алгоритмической части.

Основные источники информации:

1 Çela, E. The Quadratic Assignment Problem: Theory and Algorithms / E. Çela. — Dordrecht : Springer, 1998. — 287 p.

2 Papadimitriou, C. H. Combinatorial Optimization: Algorithms and Complexity / C. H. Papadimitriou, K. Steiglitz. — Mineola : Dover Publications, 1998. — 496 p.

3 Cormen, T. H. Introduction to Algorithms / T. H. Cormen, C. E. Leiserson, R. L. Rivest, C. Stein. — 4th ed. — Cambridge : MIT Press, 2022. — 1312 p.

4 Local Search in Combinatorial Optimization / ed. by E. Aarts, J. K. Lenstra. — Princeton : Princeton University Press, 2003. — 528 p.

5 Kirkpatrick, S. Optimization by Simulated Annealing / S. Kirkpatrick, C. D. Gelatt, M. P. Vecchi // Science. — 1983. — Vol. 220, № 4598. — P. 671–680.

6 Holland, J. H. Adaptation in Natural and Artificial Systems / J. H. Holland. — Ann Arbor : University of Michigan Press, 1975. — 183 p.

7 Levenshtein, V. I. Binary Codes Capable of Correcting Deletions, Insertions, and Reversals / V. I. Levenshtein // Soviet Physics Doklady. — 1966. — Vol. 10, № 8. — P. 707–710.