# Экспертный анализ математической постановки задачи составления школьного расписания

## 1. Введение: Комбинаторная природа задачи составления расписаний

### 1.1. Постановка проблемы и ее сложность

Задача составления расписания является классической проблемой в области комбинаторной оптимизации и планирования ресурсов, которая заключается в эффективном распределении ограниченных ресурсов (преподавателей, аудиторий) по временным интервалам для проведения запланированных занятий.1 Основными сущностями в этой задаче являются

**события** (лекции, уроки), **ресурсы** (учителя, учебные группы, кабинеты) и **временные интервалы** (дни недели, часы, пары).2 Цель состоит в том, чтобы найти такое назначение событий ресурсам и времени, которое удовлетворяет определенному набору требований.1

Исторически ранние подходы к решению этой задачи носили характер «проб и ошибок» и не опирались на строгую математическую теорию. Однако со временем стало очевидно, что проблема может быть сформулирована в точных математических терминах.5 Тем не менее, даже при самой точной формулировке, задача остается чрезвычайно сложной с вычислительной точки зрения. Общепризнано, что задача составления расписания относится к классу

**NP-трудных (NP-hard)** или **NP-полных (NP-complete)** проблем.1 Это означает, что для ее решения не существует алгоритма, который бы гарантировал нахождение оптимального решения за полиномиальное время (т.е. за время, ограниченное полиномом от размера входных данных) для всех возможных случаев. В связи с этим для крупных и реалистичных задач поиск гарантированно оптимального решения может быть вычислительно невыполнимым.9

### 1.2. Опровержение концепции «готовой» постановки

Поиск «готовой» математической постановки задачи составления расписания представляет собой распространенное заблуждение. Тщательный анализ проблемы показывает, что универсальной, готовой к использованию модели, подходящей для любого учебного заведения, не существует. Эта особенность проистекает из двух фундаментальных причин.

Во-первых, каждая реальная задача обладает уникальными, специфическими для конкретного контекста условиями.11 Требования к расписанию в одном учебном заведении могут существенно отличаться от требований в другом, что обусловлено вариативностью учебных планов, доступностью ресурсов и локальными правилами. Попытки создать точную математическую модель для конкретного университета или школы требуют детального анализа условий реализации задачи, выбора адекватного инструмента и формирования индивидуального массива исходных данных.11 Таким образом, вместо поиска готовой модели, основной задачей становится разработка гибкого подхода к ее конструированию.

Во-вторых, в реальных условиях задача часто является **«сверх-ограниченной» (over-constrained)**.12 Это означает, что не все требования и пожелания (особенно преподавателей и учащихся) могут быть удовлетворены одновременно. В таких случаях невозможно найти идеальное решение, которое бы выполняло все условия. Вместо этого необходимо найти компромиссное, но в то же время эффективное решение.5 Эта особенность требует разделения всех правил на две категории:

**жесткие ограничения**, которые не могут быть нарушены, и **мягкие ограничения**, которые могут быть нарушены, но с определенным «штрафом».12 Это разделение является краеугольным камнем современной математической постановки.

Таким образом, истинная ценность заключается не в получении заранее написанной формулы, а в понимании принципов ее построения, что позволяет адаптировать модель под конкретные потребности. Настоящий экспертный подход к проблеме расписания состоит в умении формализовать уникальный набор ограничений и критериев, а также в выборе подходящего математического аппарата для их обработки.

## 2. Основные элементы математической модели: сущности, множества и переменные

Построение математической модели начинается с формального определения всех задействованных элементов. Это ключевой этап, поскольку он создает основу для последующего формирования ограничений и целевой функции.

### 2.1. Определение ключевых сущностей и их формализация

Задача составления расписания сводится к назначению событий (уроков) определенным ресурсам (преподавателям, аудиториям, группам) в конкретные временные интервалы. Эти сущности можно формализовать с помощью математических множеств:

* **События (Events)**: Уроки, лекции или другие учебные занятия. Каждое событие e принадлежит к множеству всех событий E. Событие характеризуется продолжительностью De​∈N.3
* **Ресурсы (Resources)**: Все, что необходимо для проведения занятия.
  + **Преподаватели (Teachers)**: Множество всех преподавателей I={i1​,i2​,…,im​}.3
  + **Учебные группы (Classes/Student groups)**: Множество всех групп учеников, которые посещают занятия, G={g1​,g2​,…,gp​}.3 В более простой модели они могут быть обозначены как множество  
    J.16
  + **Аудитории (Rooms/Classrooms)**: Множество доступных аудиторий R={r1​,r2​,…,rq​}.3
* **Временные интервалы (Time Periods)**: Учебный день обычно делится на дискретные временные слоты (пары, уроки). Множество всех доступных временных слотов K={k1​,k2​,…,kl​}.3

### 2.2. Формализация переменных решения

Для построения формальной математической модели, особенно в рамках целочисленного программирования, необходимо определить **переменные решения**. Эти переменные будут принимать значения, которые представляют собой итоговое расписание. Наиболее распространенный подход заключается в использовании бинарных переменных, которые принимают значение 1, если определенное назначение выполнено, и 0 — в противном случае.16

Самая полная и часто используемая форма переменной решения — это тройная бинарная переменная, которая связывает преподавателя, занятие и время:

* Xijk​∈{0,1}, где Xijk​=1, если преподаватель i∈I ведет занятие (курс) j∈J в течение временного интервала k∈K. В противном случае, Xijk​=0.16

Этот тип переменной позволяет легко формализовать большинство жестких ограничений. Однако в некоторых случаях могут использоваться и другие формы переменных для упрощения модели 3:

* X(d,mj)​∈{0,1}, если занятие mj​ назначено на день d.
* Y(t,mj)​∈{0,1}, если занятие mj​ назначено на время t.

Выбор переменных напрямую влияет на сложность модели и алгоритма, используемого для ее решения. Использование бинарных переменных позволяет легко выразить ограничения типа «или/или», что является основой большинства задач планирования.

### 2.3. Спецификация данных (Параметры)

В дополнение к множествам и переменным, модель требует набора входных параметров, которые описывают статические данные проблемы 16:

* Ji​⊆J: множество курсов, которые может вести преподаватель i∈I.16
* Li​: максимальная нагрузка (количество курсов) для преподавателя i∈I.16
* Ck​: количество доступных аудиторий в течение временного интервала k∈K.16
* PCij​: значение, отражающее предпочтение преподавателя i к назначению на курс j.16
* PTik​: значение, отражающее предпочтение преподавателя i к временному интервалу k.16

Эти параметры, в сочетании с переменными, позволяют построить полную математическую модель.

## 3. Моделирование ограничений: обязательные и предпочтительные правила

Ограничения являются сердцем любой математической модели расписания. Их разделение на жесткие и мягкие является ключевым концептуальным моментом, который позволяет обрабатывать как обязательные требования, так и желательные предпочтения.

### 3.1. Жесткие ограничения (Hard Constraints)

Жесткие ограничения — это правила, которые должны быть выполнены для того, чтобы расписание считалось допустимым.12 Нарушение любого жесткого ограничения делает решение некорректным. Они обычно формализуются как строгие равенства или неравенства.

На основе представленных материалов, можно выделить следующие примеры жестких ограничений, выраженных с использованием бинарных переменных Xijk​:

* **Ограничение по преподавателям:** Преподаватель может проводить только одно занятие в один момент времени.
  + ∑j∈Ji​​Xijk​≤1,∀i∈I,k∈K.16
* **Ограничение по учебным группам:** Ученики одной группы не могут находиться в двух разных местах одновременно.
  + ∑i∈I​Xijk​≤1,∀j∈J,k∈K (по аналогии с ограничением для преподавателей).18
* **Ограничение по аудиториям:** В одной аудитории может проходить только одно занятие одновременно.
  + Xrjk​≤1 (для каждого занятия j и аудитории r).18
* **Ограничение на максимальную нагрузку:** Каждый преподаватель должен вести как минимум один курс, но не более своей максимальной нагрузки Li​.
  + 1≤∑j∈Ji​​∑k∈K​Xijk​≤Li​,∀i∈I.16
* **Ограничение на назначение:** Каждое занятие должно быть назначено преподавателю в определенный временной интервал.
  + ∑i∈I​∑k∈K​Xijk​=1,∀j∈J.16
* **Ограничения на соответствие:** Преподаватели могут вести только те курсы, которые им назначены, и занятия должны проходить в подходящих по вместимости и оборудованию аудиториях.15

### 3.2. Мягкие ограничения (Soft Constraints)

Мягкие ограничения являются предпочтительными, но не обязательными для выполнения.12 Их нарушение не делает расписание недействительным, но ухудшает его «качество». Выполнение мягких ограничений является целью оптимизации, и они, как правило, преобразуются в члены целевой функции, за нарушение которых начисляются штрафы.13

Примеры мягких ограничений:

* **Минимизация «окон»:** Желательно исключить свободное время между уроками как для преподавателей, так и для студентов.1 Это способствует оптимизации рабочего дня и снижению времени простоя.19
* **Равномерность нагрузки:** Рекомендуется равномерно распределять количество уроков по дням недели, например, 5/5 предпочтительнее, чем 4/6.18
* **Предпочтения преподавателей:** Учитываются личные пожелания преподавателей по времени или дням проведения занятий.18 В одной из моделей это формализовано через параметры  
  PCij​ (предпочтение к курсу) и PTik​ (предпочтение ко времени), которые используются в целевой функции.16
* **Создание «блоков» занятий:** Предпочтительно объединять несколько уроков по одному и тому же предмету в один «блок» (например, две последовательные пары), так как это может способствовать улучшению учебного процесса.19

### 3.3. Таблица «Жесткие vs. Мягкие ограничения»

|  |  |  |  |
| --- | --- | --- | --- |
| **Тип ограничения** | **Характеристика** | **Примеры** | **Обработка в модели** |
| **Жесткие (Hard)** | Обязательные условия, которые должны быть выполнены для получения корректного решения. Нарушение приводит к недействительному расписанию. | Преподаватель может вести только один урок в одно и то же время; в одной аудитории может быть только один урок; у одной группы не может быть двух занятий одновременно.12 | Формализуются как обязательные неравенства или равенства в модели. |
| **Мягкие (Soft)** | Желательные, но не обязательные условия, выполнение которых улучшает качество расписания. Нарушение не делает решение недействительным. | Устранение «окон» между уроками; равномерное распределение нагрузки; учет предпочтений преподавателей по времени; минимизация времени на перемещение между корпусами.1 | Преобразуются в члены целевой функции, за нарушение которых начисляются штрафы. |

## 4. Сравнительный анализ математических моделей

Для решения задачи составления расписания исторически применялись и развивались несколько математических подходов. Каждый из них имеет свои сильные и слабые стороны, что определяет их применимость в зависимости от характера задачи.

### 4.1. Модель на основе целочисленного программирования (Integer Programming, IP)

**Описание.** Это один из наиболее мощных и выразительных подходов. Он формулирует задачу как проблему оптимизации, в которой необходимо максимизировать или минимизировать целевую функцию, состоящую из линейных выражений, при соблюдении набора линейных неравенств (ограничений).4 Переменные решения в таких моделях являются целочисленными, чаще всего бинарными.

**Пример постановки.** Модель из источника 16 (A Mathematical Programming Model For A Timetabling Problem) является ярким примером такого подхода. Она комбинирует назначение преподавателей и планирование курсов в единую комплексную модель.

Целевая функция (Objective Function) направлена на максимизацию предпочтений:

$$ \text{Maximize } \sum\_{i \in I} \sum\_{j \in J\_i} \sum\_{k \in K} (PC\_{ij} \times X\_{ijk} + PT\_{ik} \times X\_{ijk}) $$

где PCij​ — предпочтение преподавателя i к курсу j, а PTik​ — предпочтение преподавателя i к временному интервалу k.16 Эта функция позволяет свести различные предпочтения в один критерий, который необходимо максимизировать. Модель также включает набор жестких ограничений, описанных в разделе 3.1, которые должны быть удовлетворены для получения допустимого решения.16

**Преимущества.** Подход IP обладает высокой выразительной способностью, позволяя точно формализовать сложные, многокритериальные целевые функции, которые представляют собой взвешенную сумму различных критериев.11

**Недостатки.** Главный недостаток — это высокая вычислительная сложность. Решение крупномасштабных задач с помощью точных методов, таких как метод ветвей и границ, может оказаться крайне медленным и непрактичным.4

### 4.2. Модель на основе теории графов (Graph Theory)

**Описание.** Этот подход является одним из самых ранних и известных. Задача расписания моделируется как проблема раскраски графа.8

Формализация. В этой модели:

* **Вершины (Nodes)** графа представляют собой события (уроки).
* **Ребра (Edges)** соединяют две вершины (события), если они конфликтуют и не могут быть назначены на один и тот же временной интервал (например, два урока, требующие одного и того же преподавателя или аудитории).8
* Цвета (Colors) соответствуют временным слотам.  
  Цель состоит в том, чтобы найти минимальное число цветов, необходимых для раскраски графа, чтобы никакие две смежные вершины (связанные ребром) не имели одинаковый цвет.21 Минимальное число цветов называется  
  **хроматическим числом** графа.

**Ограничения и недостатки.** Модель на основе теории графов эффективна для моделирования жестких ограничений, связанных с конфликтами ресурсов. Однако она плохо справляется с мягкими ограничениями, такими как минимизация окон или равномерность нагрузки, поскольку эти критерии не могут быть выражены простыми конфликтами.8 Это объясняет, почему этот подход редко используется сам по себе для решения сложных задач и часто служит частью более широкого гибридного алгоритма.8

### 4.3. Модель на основе задачи удовлетворения ограничений (Constraint Satisfaction Problem, CSP)

**Описание.** В отличие от IP, которая фокусируется на оптимизации, подход CSP является декларативным и направлен на поиск **допустимого** решения, которое удовлетворяет всем жестким ограничениям.23

Ключевые элементы. Модель CSP определяется тремя основными компонентами 23:

* **Переменные (Variables)**: Каждое событие, которое необходимо запланировать.
* **Домены (Domains)**: Множество всех возможных значений для каждой переменной, например, все доступные временные слоты для данного урока.24
* **Ограничения (Constraints)**: Набор правил, которые должны быть выполнены. Они делятся на жесткие и мягкие.24

**Сравнение с IP.** CSP более эффективен в обработке жестких ограничений за счет механизма **«прогонки ограничений» (constraint propagation)**, который активно сужает пространство поиска, исключая заведомо недействительные назначения.13 Это делает его особенно подходящим для задач, где приоритетом является нахождение любого осуществимого расписания. В то же время, IP лучше подходит для оптимизационных задач, где необходимо не просто найти решение, а найти наилучшее, основываясь на числовой целевой функции.

### 4.4. Сводная таблица сравнения моделей

|  |  |  |  |
| --- | --- | --- | --- |
| **Критерий** | **Целочисленное программирование (IP)** | **Теория графов** | **Задача удовлетворения ограничений (CSP)** |
| **Основная цель** | Оптимизация (минимум/максимум целевой функции). | Минимизация количества необходимых ресурсов (временных слотов). | Поиск допустимого решения, удовлетворяющего всем жестким ограничениям. |
| **Переменные** | Бинарные (Xijk​) или другие целочисленные переменные. | Вершины графа. | Переменные, представляющие события. |
| **Целевая функция** | Явно заданная линейная или нелинейная функция. | Хроматическое число графа. | Отсутствует в чистом виде; оптимизация осуществляется путем минимизации штрафов за нарушение мягких ограничений. |
| **Обработка ограничений** | Жесткие: линейные неравенства. Мягкие: члены целевой функции. | Только жесткие: выражаются как ребра-конфликты. | Жесткие: прогонка ограничений. Мягкие: вес/стоимость нарушения. |
| **Преимущества** | Высокая выразительность; мощный аппарат для многокритериальной оптимизации. | Наглядная визуализация конфликтов; эффективен для базовых проблем без мягких ограничений. | Декларативный подход; эффективная прогонка ограничений; хорошо обрабатывает сложные жесткие ограничения. |
| **Недостатки** | Высокая вычислительная сложность для крупных задач. | Плохо справляется с мягкими ограничениями. | Менее эффективен для сложных оптимизационных задач. |

## 5. Целевая функция: критерии оптимальности и многокритериальная оптимизация

### 5.1. Роль целевой функции

Целевая функция является неотъемлемой частью любой оптимизационной модели. Ее роль заключается в преобразовании набора желательных, но не обязательных условий (мягких ограничений) в единый числовой критерий, который алгоритм должен оптимизировать.12 Это позволяет сравнивать различные допустимые расписания между собой и выбирать из них наиболее качественное.

В общем виде целевая функция представляет собой взвешенную сумму штрафов за нарушение мягких ограничений, которую необходимо минимизировать, или взвешенную сумму выгод от их выполнения, которую необходимо максимизировать. Типичная форма для минимизации:

Minimize i∑​wi​×penaltyi​

где wi​ — весовой коэффициент, отражающий важность i-го мягкого ограничения, а penaltyi​ — числовой штраф за его нарушение.12

### 5.2. Примеры критериев оптимизации

Анализ показывает, что не существует одного «наиболее важного» критерия для оптимизации расписания.20 Набор критериев зависит от заинтересованных сторон (администрации, преподавателей, студентов) и их приоритетов. Это делает оптимизацию расписания многокритериальной задачей, где оптимальное решение для одного участника может быть неоптимальным для другого.

Примеры критериев, которые могут быть включены в целевую функцию:

* **Минимизация конфликтов:** Простейшая целевая функция, которая просто суммирует количество всех нарушений, будь то конфликты ресурсов или несоблюдение предпочтений.25
* **Максимизация предпочтений:** Учет предпочтений преподавателей по времени и курсам. Например, модель может максимизировать сумму предпочтений PCij​ и PTik​, как это было показано в разделе 4.1.16
* **Оптимизация учебного процесса:** Создание «блоков» из нескольких последовательных занятий по одному предмету. Целевая функция может быть направлена на максимизацию количества таких блоков, поскольку это может улучшить процесс обучения.19
* **Равномерность и баланс:** Обеспечение равномерного распределения учебной нагрузки по дням недели для групп и преподавателей. Это может быть выражено как минимизация дисперсии числа занятий в день.20

## 6. Методы решения: от точных до эвристических

### 6.1. Почему точные методы часто неприменимы

Как уже отмечалось, NP-трудность проблемы составления расписания делает невозможным использование точных алгоритмов для поиска глобально оптимального решения в приемлемые сроки, особенно для крупномасштабных, реальных экземпляров.6 Точные методы, такие как целочисленное программирование, могут найти оптимальное решение для небольших задач, но их вычислительная сложность быстро возрастает с увеличением числа переменных и ограничений.4 Таким образом, для практических целей необходимо использовать подходы, которые жертвуют абсолютной оптимальностью ради вычислительной эффективности.

### 6.2. Метаэвристические подходы

Метаэвристики — это приближенные алгоритмы, которые не гарантируют нахождение глобального оптимума, но способны найти «достаточно хорошие» (близкие к оптимальным) решения за разумное время.26 Они особенно эффективны для задач с большим пространством поиска.

* **Генетические алгоритмы (Genetic Algorithms, GA):**
  + **Принципы:** Основаны на эволюционной биологии. Алгоритм работает с популяцией потенциальных решений («особей»), каждое из которых представляет собой возможное расписание. «Приспособленность» (fitness) каждого расписания оценивается на основе целевой функции (например, количеством разрешенных конфликтов).10 На каждой итерации (поколении) отбираются лучшие решения, которые затем «скрещиваются» (crossover) и «мутируют» (mutation), создавая новые, более качественные решения.27
  + **Преимущества:** GA хорошо подходят для решения сложных задач с большим количеством переменных и ограничений, способны находить качественные решения.
* **Имитация отжига (Simulated Annealing, SA):**
  + **Принципы:** Аналогия с физическим процессом отжига металла. Алгоритм исследует пространство решений, принимая «соседние» решения. На начальных этапах («высокой температуре») он с определенной вероятностью принимает даже худшие решения, что позволяет ему избежать застревания в локальных оптимумах.25 Постепенно «температура» снижается, и алгоритм становится более избирательным, сходясь к хорошему решению.
  + **Преимущества:** Эффективен для задач с большим количеством локальных минимумов.

**Другие методы.** Существуют и другие мощные метаэвристические подходы, такие как **поиск с запретами (Tabu Search)**, а также **гибридные подходы**, которые объединяют сильные стороны различных методов, например, сочетание целочисленного программирования с метаэвристиками.9 Эти

**«матевристики»** часто показывают высокую эффективность в решении сложных реальных задач.9

### 6.3. Сводная таблица сравнения методов решения

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Метод** | **Тип** | **Скорость** | **Качество решения** | **Применимость** |
| **Целочисленное программирование (IP)** | Точный | Медленная для крупных задач | Гарантированный оптимум (при наличии решения) | Малые и средние задачи; задачи, где требуется абсолютная оптимальность. |
| **Задача удовлетворения ограничений (CSP)** | Точный | Зависит от сложности ограничений | Допустимое, но не обязательно оптимальное решение | Задачи с большим количеством жестких ограничений, где приоритет — осуществимость. |
| **Генетические алгоритмы (GA)** | Приближенный (метаэвристика) | Высокая | Близкое к оптимальному | Сложные, крупномасштабные задачи, где точное решение не требуется. |
| **Имитация отжига (SA)** | Приближенный (метаэвристика) | Высокая | Близкое к оптимальному; избегает локальных минимумов | Задачи с большим количеством локальных оптимумов. |

## 7. Заключение и выводы

Проведенный анализ показывает, что задача составления школьного расписания является глубоко NP-трудной проблемой, что делает невозможным существование единой «готовой» математической формулировки, применимой во всех случаях. Фундаментальная причина этого заключается в уникальности ограничений и предпочтений каждого учебного заведения, а также в «сверх-ограниченном» характере реальных задач.

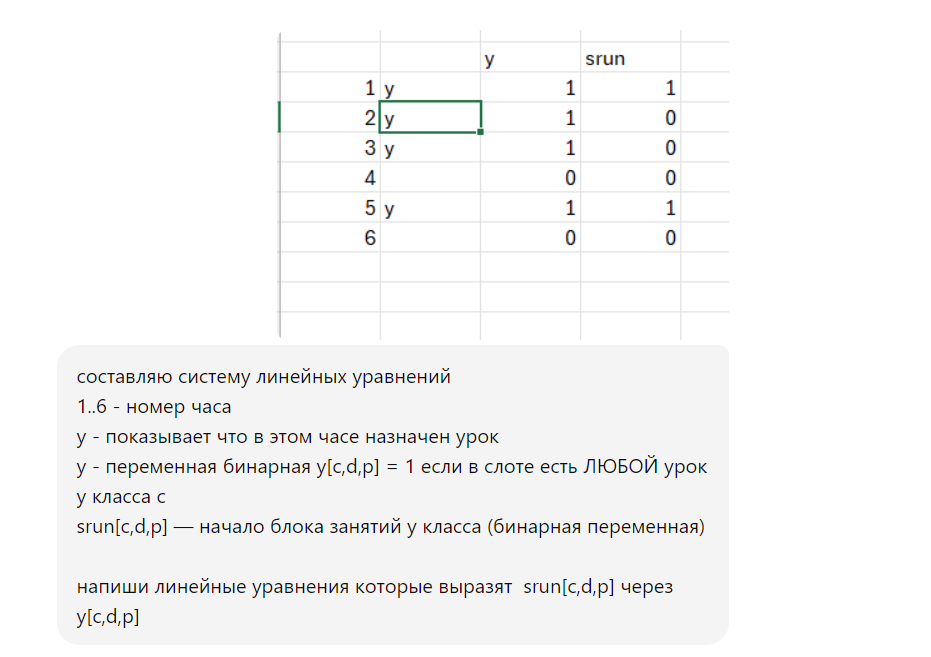
**Ключевые выводы:**

1. **Отсутствие универсальной модели:** Не существует одной-единственной готовой постановки. Вместо этого, для каждого случая требуется построение уникальной модели, основанной на наборе сущностей (события, ресурсы, время) и формализации переменных решения, чаще всего бинарных.
2. **Два типа ограничений:** Все требования должны быть четко разделены на **жесткие** (обязательные, не подлежащие нарушению) и **мягкие** (желательные, но необязательные, за нарушение которых начисляется штраф). Это разделение является основой для любой оптимизационной модели.
3. **Выбор модели:** Выбор математического аппарата зависит от приоритетов. Для задач, где необходимо найти гарантированно оптимальное решение и можно точно сформулировать целевую функцию, подходит **целочисленное программирование (IP)**. Если основная цель — найти любое допустимое решение, удовлетворяющее большому количеству жестких ограничений, более эффективен подход на основе **задачи удовлетворения ограничений (CSP)**.
4. **Практические решения:** Для решения крупных, реалистичных задач, где точные методы оказываются слишком медленными, необходимо использовать **метаэвристические алгоритмы** (например, генетические алгоритмы или имитация отжига). Эти методы жертвуют абсолютной оптимальностью ради практической скорости и находят решения, которые являются «достаточно хорошими» для использования в реальном мире.

Таким образом, для успешного решения задачи составления расписания необходимо не просто найти готовую формулу, а глубоко понимать ее структуру, правильно определить ограничения и выбрать наиболее подходящий вычислительный метод для их обработки. Будущие направления исследований в этой области включают разработку гибридных методов, которые комбинируют точные подходы с эвристиками, а также развитие систем многокритериальной оптимизации, способных учитывать интересы всех заинтересованных сторон.

#### Works cited

1. Decomposing the High School Timetable Problem, accessed August 27, 2025, <https://www.patatconference.org/patat2012/proceedings/2_14.pdf>
2. Задача составления расписаний: решение на основе ..., accessed August 27, 2025, <https://cyberleninka.ru/article/n/zadacha-sostavleniya-raspisaniy-reshenie-na-osnove-mnogoagentnogo-podhoda>
3. Mathematical Modelling of University Timetabling: A Mathematical Programming Approach, accessed August 27, 2025, <https://www.researchgate.net/publication/278330149_Mathematical_Modelling_of_University_Timetabling_A_Mathematical_Programming_Approach>
4. A Mixed-Integer Linear Programming Model for University Course Timetabling Problems - WUR eDepot, accessed August 27, 2025, <https://edepot.wur.nl/637835>
5. Mathematical investigation of a model of the school timetabling problem - Aston Research Explorer, accessed August 27, 2025, <https://research.aston.ac.uk/en/studentTheses/mathematical-investigation-of-a-model-of-the-school-timetabling-p>
6. Задача об оптимальном планировании работы - Википедия, accessed August 27, 2025, <https://ru.wikipedia.org/wiki/%D0%97%D0%B0%D0%B4%D0%B0%D1%87%D0%B0_%D0%BE%D0%B1_%D0%BE%D0%BF%D1%82%D0%B8%D0%BC%D0%B0%D0%BB%D1%8C%D0%BD%D0%BE%D0%BC_%D0%BF%D0%BB%D0%B0%D0%BD%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B8_%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D1%8B>
7. АЛГОРИТМЫ КОМБИНАТОРНОЙ ОПТИМИЗАЦИИ, СОЧЕТАЮЩИЕ ЖАДНЫЕ СТРАТЕГИИ И О, accessed August 27, 2025, <https://asvk.cs.msu.ru/wp-content/uploads/2023/04/TiSU_2017_2_ZhA_perebor_rus.pdf>
8. A Study on Course Timetable Scheduling using Graph Coloring Approach - Research India Publications, accessed August 27, 2025, <https://www.ripublication.com/ijcam17/ijcamv12n2_26.pdf>
9. Timetabling at High Schools - DTU Research Database, accessed August 27, 2025, <https://orbit.dtu.dk/files/99520418/Timetabling_at_High_Schools.pdf>
10. A comparative study on solving university timetabling problems with emphasis on Genetic Algorithms - International Association for Computer Information Systems, accessed August 27, 2025, <https://iacis.org/iis/2024/4_iis_2024_392-408.pdf>
11. Математическое моделирование задачи составления ..., accessed August 27, 2025, <https://cyberleninka.ru/article/n/matematicheskoe-modelirovanie-zadachi-sostavleniya-raspisaniya-zanyatiyvuza>
12. University Timetabling using Constraint Handling Rules - CiteSeerX, accessed August 27, 2025, <https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=9df9a6b09fb35835cf58aad9430b56638c16d490>
13. University Course Timetabling with Soft Constraints - UniTime, accessed August 27, 2025, <https://www.unitime.org/papers/patat03.pdf>
14. Integer programming for the generalized high school timetabling ..., accessed August 27, 2025, <https://orbit.dtu.dk/files/128525231/Integer_Programming_MIP_XHSTT_journal_of_scheduling.pdf>
15. Interactive Timetabling - UniTime, accessed August 27, 2025, <https://www.unitime.org/papers/it02_ctp.pdf>
16. (PDF) A Mathematical Programming Model For A Timetabling ..., accessed August 27, 2025, <https://www.researchgate.net/publication/221143006_A_Mathematical_Programming_Model_For_A_Timetabling_Problem>
17. An Integer Programming Model for the School Timetabling Problem - LAC/INPE, accessed August 27, 2025, <http://www.lac.inpe.br/~lorena/geraldo/ribeiro-lorena-claio.pdf>
18. жесткие и мягкие ограничения в алгоритме составления ..., accessed August 27, 2025, <https://libeldoc.bsuir.by/bitstream/123456789/36590/1/Apal_Zhestkiye.pdf>
19. A mathematical programming model for high school timetabling problem | Ingeniería y Ciencia - Revistas Universidad EAFIT, accessed August 27, 2025, <https://publicaciones.eafit.edu.co/index.php/ingciencia/article/view/3124>
20. Модели и методы многокритериальной оптимизации ..., accessed August 27, 2025, <https://cyberleninka.ru/article/n/modeli-i-metody-mnogokriterialnoy-optimizatsii-raspisaniy-vuza>
21. A graph coloring algorithm for large scheduling problems, accessed August 27, 2025, <https://nvlpubs.nist.gov/nistpubs/jres/84/jresv84n6p489_a1b.pdf>
22. Timetable Scheduling using Bipartite Graph - IJRASET, accessed August 27, 2025, <https://www.ijraset.com/research-paper/timetable-scheduling-using-bipartite-graph>
23. Solving the timetabling problem using constraint satisfaction ..., accessed August 27, 2025, <https://ro.uow.edu.au/articles/thesis/Solving_the_timetabling_problem_using_constraint_satisfaction_programming/27827784>
24. Course scheduling with CSP - Seoyeong Park, accessed August 27, 2025, <https://seoyeongpark.github.io/projects/courseschedulingcsp/>
25. Constructing School Timetables Using Simulated Annealing: Sequential and Parallel Algorithms - PubsOnLine, accessed August 27, 2025, <https://pubsonline.informs.org/doi/pdf/10.1287/mnsc.37.1.98>
26. Solving real school timetabling problems with meta-heuristics - CiteSeerX, accessed August 27, 2025, <https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=ee89cce232b6187488db8c623bfbf8b1defd66cb>
27. time table scheduling using genetic algorithm - International Journal of Research in Advent Technology, accessed August 27, 2025, <https://ijrat.org/downloads/Vol-1/oct-2013/paper%20id-13201322.pdf>



Изображение выглядит как текст, Шрифт, снимок экрана, документ

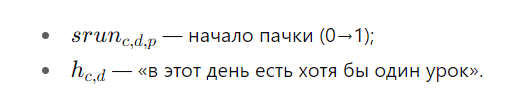
Содержимое, созданное искусственным интеллектом, может быть неверным.

мне надо посчитать количество "окон" между уроками zgap в данном примере оно равно 1 вырази zgap через y с помощью линейных уравнений

Окно» — это нулевая пачка между двумя пачками единиц.  
Тогда число окон за день равно числу пачек занятий минус 1 (если в день вообще есть занятия).

Окно» — это нулевая пачка между двумя пачками единиц.  
Тогда число окон за день равно числу пачек занятий минус 1 (если в день вообще есть занятия).

Пусть p=1,…,Pp=1,\dots,Pp=1,…,P — слоты, yc,d,p∈{0,1}y\_{c,d,p}\in\{0,1\}yc,d,p​∈{0,1} — «в этом слоте есть ЛЮБОЙ урок».  
Введём вспомогательные бинарные переменные:



Изображение выглядит как текст, Шрифт, снимок экрана, рукописный текст

Содержимое, созданное искусственным интеллектом, может быть неверным.