Министерство науки и высшего образования Российской Федерации
Санкт-Петербургский политехнический университет Петра Великого
Институт компьютерных наук и технологий

Работа допущена к защите
директор ВШИСиСТ
_____ А.В. Щукин
«_____» _____ 2021 г.

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА
РАБОТА БАКАЛАВРА
РАЗРАБОТКА СИСТЕМЫ СОСТАВЛЕНИЯ ПРЕДВАРИТЕЛЬНОГО РАСПИСАНИЯ СЕССИИ

по направлению подготовки 09.03.03 Прикладная информатика
Направленность (профиль) 09.03.03_YY Наименование направленности (профиля) образовательной программы

Выполнил
студент гр. 3530903/70302 Д.В. Сухова

Руководитель
директор ВШИСиСТ,
к.н.т., старший преподаватель А.В. Щукин

Консультант
ассистент ВШИСиСТ В.А. Пархоменко

Консультант
по нормоконтролю[1] В.А. Пархоменко

Санкт-Петербург
2021

[1]

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Данный пример выпускной квалификационной работы (далее — ВКР) создан для того, чтобы продемонстрировать возможности шаблонов SPbPU–student-templates, выполненных с помощью издательской системы LaTeX [**spbpu-student-thesis-template**]. В примере отображены некоторые обязательные элементы ВКР [**spbpu-student-thesis-specification**]. Для того, чтобы подробнее ознакомиться с требованиями к наполнению этих элементов, а также с общими требованиями к структуре и оформлению ВКР, пожалуйста, ознакомьтесь с [**spbpu-student-thesis-template-author-guide**; **spbpu-student-thesis-specification**].

Пример может быть использован для подготовки отчета по практике с учетом корректировки требований к титульному листу, структуре и содержанию конкретной практики в соответствии с планом её проведения (рабочей программы дисциплины).

Технология написания ВКР на LaTeX подробно изложена в [**spbpu-student-thesis-template-author-guide**]. В рекомендациях приведены ссылки на учебно-справочные материалы LaTeX (под LaTeX в документе может подразумеваться также TeX, LaTeX 2ε).

Авторам, использующим LaTeX, необходимо последовательно заменять текст данного шаблона в файлах «`thesis.tex`» на текст своей ВКР, избегая при этом ошибок (errors) при компиляции. Синтаксические конструкции LaTeX, которые задействованы в формировании того или иного текста выделены `машинописным шрифтом`. Иные шрифты в тексте ВКР (за исключением математических) использовать запрещено.

Светлым курсивом выделены *важные* элементы текста (ключевые слова определений, интонационные выделения словосочетаний), полужирным шрифтом — **служебные** элементы текста («определение», «теорема», «лемма» и т.п), а также при необходимости ключевые слова в алгоритмах. В соотношении к основному тексту курсив и полужирный шрифт не может превышать 1 % текста на странице.

Полужирный курсив разрешено использовать только в ***названиях подпараграфов (пунктов)*** и запрещёно использовать в основном тексте. Подчеркивание допускается использовать только в задании в местах, где данные вписываются студентом, а также в математических формулах при необходимости.

Введение *не должно превышать 4 страницы*. Во введении необходимо обосновать выбор темы, охарактеризовать современное состояние изучаемой проблемы, ее актуальность, практическую и теоретическую значимость, степень разработанности данной проблемы.

Актуальность исследования заключается в N фактах и явлениях, а также в их состоянии, связанных с ними нерешенных проблемах, слабо освещенных и требующих уточнения или дальнейшей разработки вопросов.

Объект исследования — это то, на что направлен процесс познания (индивид, коллектив, общность людей, сфера деятельности и т.п.). Связь объекта и предмета легко запоминаются по формуле: «исследуем такой-то объект на предмет чего-то». Это процесс или явление, порождающее проблемную ситуацию, и избран-ное для изучения в целом. Всегда в объекте содержится предмет, а не наоборот.

Предмет исследования — один из аспектов, часть рассматриваемого объекта (свой-ства, состояния, процессы, направления и особенности деятельности структур по связям с общественностью, их сотрудников в конкретных сферах общественных отношений и т.д.). Предмет исследования частично совпадает с названием работы и содержится в цели сразу после сказуемого («выявить . . . что?», «определить . . . что?», «сформировать . . . что?»). Именно предмет исследования определяет тему выпускной квалификационной работы. Объект и предмет исследования соотносятся между собой как целое и частное, общее и частности.

Цель исследования формулируется, исходя из проблемы, которую следует разрешить студенту в процессе выполнения выпускной квалификационной работы и представляет собой в самом сжатом виде тот результат (результаты), который должен быть получен в итоге исследования. Формулировку цели рекомендуется начинать со слов: «сформировать/создать», «разработать», «провести», «подготовить».

Цель исследования — краткий ожидаемый результат, то есть решение практических задач и новые знания о рассматриваемом предмете исследования. В соответствии с целью исследования, логически определяются следующие **задачи работы** (должно быть *не менее четырех задач, но не более шести задач*):

A. Первая задача.
B. Вторая задача.
C. Третья задача.
D. Четвертая задача.

Задачи отражают *поэтапное достижение цели, при этом уточняют границы проводимого исследования*. Рекомендуется формулировать задачи с глаголов в форме перечисления: «изучить . . . », «выявить . . . », «проанализировать . . . », «разработать . . . », «описать . . . » и т.п. Заголовки выпускной квалификационной работы должны отражать суть поставленной задачи.

Общая направленность исследования задается до его начала сформулированными **гипотезами**, которыми могут быть:

- научное предположение, выдвигаемое для объяснения каких-либо факторов, явлений и процессов, которые надо подтвердить или опровергнуть (т. е. требующее верификации);
- вероятностное знание, научно обоснованная догадка по объяснению действительности;
- прогноз ожидаемого решения проблемы, ответ на вопрос, поставленный в задаче;
- условно-категорическое умозаключение по схеме «если . . . , то . . . », основными элементами которого являются условие (причина) и результат (следствие).

Гипотеза — это предполагаемое решение проблемы. В ходе исследования гипотезу проверяют и либо подтверждают, либо опровергают. Формулировка гипотезы *обязательна только для магистров*.

Теоретическая и методологическая база исследования. В теоретической базе необходимо перечислить источники, которые использовались для написания работы. Приведём примеры ключевых фраз:

- «Теоретической основой выпускной квалификационной работы послужили исследования . . . (перечисляются конкретные документы)».
- «Практическая часть работы выполнялась на основании документов . . . ».
- «При написании выпускной квалификационной работы использовалась работы отечественных и зарубежных специалистов . . . ».
- «Для выполнения анализа в практической части были использованы материалы . . . ».
- «При подготовке ВКР были использованы материалы таких учебных дисциплин, как "Технология конструкционных материалов”, "Экономика” "Начертательная геометрия” . . . ».
- «При выполнении ВКР использовались материалы N организации . . . (ссылка на официальный сайт)».

Методологическая база исследования должна содержать указание на методы и подходы, на которых основывается данная ВКР.

Среди методов исследования студенту необходимо обратить внимание на общенаучные методы, включающие эмпирические (наблюдение, эксперимент, сравнение, описание, измерение), теоретические (формализация, аксиоматический, гипотетико-дедуктивный, восхождение от абстрактного к конкретному) и общелогические (анализ, абстрагирование, обобщение, идеализация, индукция, аналогия, моделирование и др.) методы. Также следует назвать конкретно-научные (частные) методы научного познания, представляющие собой специфические методы конкретных наук: экономики, социологии, психологии, истории, логики и проч.

Информационной базой для разработки ВКР служат материалы, собранные студентом в процессе обучения в ВУЗе, в ходе прохождения учебной и производственной практик, а также во время прохождения преддипломной практики. Дополнительная информационная база может включать информацию официальных статистических публикаций (например, Госкомстата России), материалы, получаемые из Интернета, информацию международных организаций и ассоциаций.

Степень научной разработанности проблемы — это состояние теоретической разработанности проблемы, анализ работ отечественных и зарубежных авторов, исследующих эту проблему. Здесь важно подчеркнуть исторические, экономические, политические или профессиональные явления, повлиявшие на выбор темы. Также в данной части введения проводится критический обзор современного состояния и освещения исследуемой темы в научной, профессиональной литературе и СМИ, обобщаются и оцениваются точки зрения различных авторов по теме исследования.

Научная новизна выявляется в результате анализа литературных источников, уточнения концептуальных положений, обобщения опыта решения подобных проблем. Это принципиально новое знание, полученное в науке в ходе проведенного исследования (теоретические положения, впервые сформулированные и обоснованные, собственные методические рекомендации, которые можно использовать в практике). Научная новизна выпускной квалификационной работы может состоять:

- в изучении фактов и явлений с помощью специальных научных методов и междисциплинарных подходов;

- в изучении уже известного в науке явления на новом экспериментальном материале;
- в переходе от качественного описания известных в науке фактов к их точно определяемой количественной характеристике;
- в изучении известных в науке явлений и процессов более совершенными методами;
- в сопоставлении, сравнительном анализе протекания процессов и явлений;
- в изменении условий протекания изучаемых процессов;
- в уточнении категориального аппарата дисциплины, определение типологии, признаков, специфики изучаемого явления.

Практическая значимость подробно отражается в:

- практических рекомендациях или разработанном автором выпускной квалификационной работы проекте (как основная часть выпускной квалификационной работы);
- выявлении важности решения избранной проблемы для будущей деятельности магистра по выбранному направлению подготовки.

Практическая значимость выпускной квалификационной работы может заключаться в возможности:

- решения той или иной практической задачи в сфере профессиональной деятельности;
- проведения дальнейших научных исследований по теме ВКР;
- разработки конкретного проекта, направленного на интенсификацию работы исследуемой организации, предприятия.

Апробация результатов исследования включает:

- участие в конференции, семинарах и т. д.;
- публикации по теме выпускной квалификационной работы;
- применение результатов исследования в практической области;
- разработку и внедрение конкретного проекта;
- выступления на научных конференциях, симпозиумах, форумах и т.п. (*обязательно*);
- публикации студента, включенные в список использованных источников.

В силу ограниченности объема необходимо очень тщательно подойти к написанию введения, которое должно стать «визитной карточкой», кратко, но емко характеризующей работу. Во введение не включают схемы, таблицы, описания, рекомендации и т.п.

Целью первой главы, как правило, является всесторонний анализ предмета и объекта исследования, второй — разработка предложений (алгоритмов, технологий и т.п.) по улучшению какого-либо процесса, протекающих с участием предмета и объекта исследования, третьей — практическая реализация (имплементация) — предложений (алгоритмов, технологий и т.п.) в виде программного (или иного) продукта, четвертой — апробация разработанных в работе предложений и выводы целесообразности их дальнейшей разработки (использованию). Содержание глав в данном шаблоне приведено только для демонстрации возможностей LaTeX.

ГЛАВА 1. СИСТЕМЫ СОСТАВЛЕНИЯ РАСПИСАНИЯ

Проблема составления расписания не нова, и существует множество средств, призванных упростить её решение. В интернете можно найти онлайн-календари с возможностью совместного редактирования, системы управления бизнес-процессами и программы генерации расписания для разного рода предприятий. Существует отдельная ниша систем составления расписания для ВУЗов, и в параграфе 1.1 рассматриваются её русскоязычные представители, а в параграфе 1.4 - зарубежные. Далее приведено сравнение таких систем и анализ их преимуществ и недостатков для решения конкретной задачи - составления предварительного расписания сессии в университете.

1.1. Обзор российских систем составления расписания

1.1.1. 1С: ХроноГраф Расписание

[**https://1c.ru/rus/products/1c/predpr/compat/catalog/solution.jsp?SolutionID=1**

Фирма «1С», занимающаяся разработкой ПО для бизнеса и образования в качестве системы для автоматизации учебного планирования и составления расписания в разного рода организациях предлагает свою программу «1С: ХроноГраф Расписание».

«1С: ХроноГраф Расписание» позволяет

- составлять понедельное расписание организации или отдельных её подразделений;
- задавать периоды обучения с учётом нерабочих дней, каникул и разбиением на четные и нечётные недели;
- создавать черновое расписание, используя функцию «Предварительный расчёт».

Основной проблемой данной программы является несовместимость с другими платформами. «1С: ХроноГраф Расписание» - однопользовательская программа, и нельзя интегрировать её с web-приложением для возможности сбора данных напрямую от пользователей. Сложность составления расписания сессии в этой системе обуславливается также её ориентированностью на составление расписания по неделям без учёта специфики проведения аттестаций.

1.1.2. Avtor (АВТОРасписание)

[**http://avtor.bravosoft.org/**] Программа «АВТОРасписание» имеет несколько версий для различных учебных заведений: общеобразовательных школ, колледжей, техникумов, профессиональных училищ и ВУЗов. Это позволяет в подстроиться под специфику расписания конкретного типа образовательного учреждения, что является одним из её конкрурентых преимуществ.

«АВТОРасписание» имеет достаточно широкий спектр применений. Этот программный продукт позволяет

- составлять понедельное расписание для учебных групп с минимальным количеством окон;
- составлять расписание преподавателей с минимальным количеством окон;
- оптимально размещать занятия по аудиториям, учитывая их вместимость и оснащённость необходимым оборудованием;
- учитывать пожелания сотрудников к своему расписанию;
- разделять учебные группы на подгруппы;
- вносить ручные корректировки в расписание.

Преимуществом этой программы помимо прочего является возможность публиковать расписание обучающихся и преподавателей из самой системы «Автор» на сайте, внутреннем портале или на мультимедийных стендах образовательной организации. Но при этом импорт данных всё ещё производится вручную диспетчером, что не очень удобно для учебного заведения с большим штабом сотрудников, которые сами могли бы вносить свои пожелания в систему.

1.1.3. Галактика Расписание учебных занятий

[**https://galaktika-it.ru**] «Галактика Расписание учебных занятий» - часть системы управления ВУЗом той же корпорации Галактика. Этот программный продукт позволяет составлять расписание в ВУЗе, а также:

- вычислять несколько десятков показателей эффективности расписаний;
- оптимально размещать занятия по аудиториям, учитывая их вместимость и оснащённость необходимым оборудованием;
- учитывать приоритет преподавателей, учебных групп и дисциплин;
- контролировать пересечение расписаний для преподавателей, учебных групп и подгрупп во избежание «накладок»;
- контролировать длительность занятий;

– вручную бронировать аудиторный фонд;
– учитывать план изучения дисциплин для выстраивания их в правильном порядке.

«Галактика Расписание учебных занятий» - серьёзный инструмент для формирования расписания в высших учебных заведениях, учитывающий множество факторов при его составлении и имеющий удобную систему отчётности. На данный момент эта программа наиболее полно решает проблему автоматической генерации расписания российских ВУЗов, но и она не имеет интерфейса для прямого импорта пожеланий преподавателей прямо в систему. Компания «Галактика» помимо прочего предлагает техническое сопровождение своего ПО, но это учитывается при расчёте стоиости лецензии на использование программы.

1.2. Обзор зарубежных систем составления расписания

1.2.1. Apereo UniTime

[**https://www.unitime.org/**] UniTime от компании Apereo - система автоматического создания расписания западных высших учебных заведений. Она учитывает, что студенты могут выбирвать себе индивидуальный набор курсов, чего не происходит в российских ВУЗах, где обучение происходит по плану образовательных программ направлений.

UniTime даёт возможность:

– автоматически генерировать расписание курсов и экзаменов;
– минимизировать конфликты студенческих курсов;
– вносить ручные корректировки в расписание.

Эта программа имеет понятный web-интерфейс и может быть интегрирована в другую систему, но она не позволяет преподавателям вносить данные о своей занятости, чтобы учесть их при составлении расписания. Неприспособленность программы под составление расписания для групп, а не для конкретных студентов делает её менее удобной, чем российские аналоги.

1.2.2. Lantiv Scheduling Studio

[**https://scheduling-studio.lantiv.com/**] Программа «Scheduling Studio» от компании Lantiv представляет собой систему совемстной работы над расписанием и реализует следующие задачи:

- совместный доступ к редактированию расписания ВУЗа;
- оффлайн редактирование с возможностью синхронизации после появления в сети;
- цветовое выделение накладок расписания;
- составление расписания на различные временные периоды: неделя, семестр, четверть, год;
- копирование составленных элементов расписания на другие периоды.

Данный программный продукт имеет приятный и понятный интерфейс, но не имеет модуля автоматической генерации расписания, из-за чего основная часть работы всё ещё ложится на плечи диспетчеров. «Scheduling Studio» удобно использовать для составления нетривиального расписания, которое меняется от недели к неделе и плохо вписывается в шаблон школьного расписания или расписания учебных занятий ВУЗа, например. Но для составления расписани сессии требуется большая степень автоматизации, чем предлагается этим ПО.

1.3. Сравнение рассмотренных систем составления расписания

Сведения о возможностях каждого из описанного в параграфах 1.1 и 1.4 сведём в таблицу 1.1.

Таблица 1.1

Сравнение систем составления расписания

	Учитывает пожелания ППС	Интегрируется с сайтами ВУЗов	Имеет возможность задавать нетривиальное расписание	Плата за использование	Генерация предварительного расписания	Открытый исходный код
1С: ХроноГраф Расписание	+	-	-	+	+	-
Avtor	+	-	+	+	+	-
Галактика Расписание учебных занятий	+	+	+	+	+	-
Apereo UniTime	-	+	+	-	+	+
Lantiv Scheduling Studio	-	-	+	+	-	-

Видно, что среди систем составления расписания для составления предварительного расписания сессии лучше всего могла бы подойти программа «Галактика Расписание учебных занятий», так как она удовлетворяет большинству требований, но при этом она, как и почти всё представленное в таблице ПО, требует оплату за использование. Также среди представленных программных продуктов все, кроме одного, не предоставляют открытый доступ к исходному коду. Таким образом, в рамках данной работы необходимо было разработать программу, удовлетворяющую всем перечисленным в таблице критериям.

Одиночные формулы оформляют в окружении `equation`, например, как указано в следующей одиночной нумерованной формуле:

$$\pi \approx 3{,}141. \tag{1.1}$$

Рис.1.1. Вид на гидробашню СПбПУ [**spbpu-gallery**]

На рис.1.1 изображена гидробашня СПбПУ, а в табл.**??** приведены данные, на примере которых коротко и наглядно будет изложена суть ВКР.

1.4. Название параграфа

Формулы могут быть размещены в несколько строк. Чтобы выставить номер формулы напротив средней строки, используйте окружение `multlined` из пакета

`mathtools` следующим образом [**Ganter1999**]:

$$\begin{aligned}(A_1,B_1) \leqslant (A_2,B_2) \;\; &\Leftrightarrow \\ \Leftrightarrow \;\; A_1 \subseteq A_2 \;\; &\Leftrightarrow \\ \Leftrightarrow \;\; B_2 \subseteq B_1&.\end{aligned} \tag{1.2}$$

Используя команду `\labelcref` из пакета `cleveref`, допустимо следующим образом оформлять ссылку на несколько формул: (1.1 и 1.2). На рис.1.2 приведены три картинки под общим номером и названием, но с раздельной нумерацией подрисунков посредством пакета `subcaption`.

a)  b)  c) 

Рис.1.2. Фотографии Белого зала СПбПУ [**spbpu-gallery**], в том числе: *a* — со стороны зрителей; *b* — со стороны сцены; *c* — барельеф

Далее можно ссылаться на три отдельных рисунка: рис.1.2a, рис.1.2b и рис.1.2c.

Пример ссылок [**Article**; **Book**; **Booklet**; **Conference**; **Inbook**; **Incollection**; **Manual**; **Mastersthesis**; **Misc**; **Phdthesis**; **Proceedings**; **Techreport**; **Unpublished**; **badiou:briefings**], а также ссылок с указанием страниц, на котором отображены номера страниц [**Naidenova2017**] или в виде мультицитаты на несколько источников [**Naidenova2017**; **Ganter1999**]. Часть библиографических записей носит иллюстративный характер и не имеет отношения к реальной литературе.

1.5. Выводы

Текст выводов по главе 1.

Кроме названия параграфа «выводы» можно использовать (единообразно по всем главам) следующие подходы к именованию последних разделов с результатами по главам:

– «выводы по главе N», где N — номер соответствующей главы;
– «резюме»;
– «резюме по главе N», где N — номер соответствующей главы.

Параграф с изложением выводов по главе *является обязательным*.

ГЛАВА 2. АЛГОРИТМЫ СОСТАВЛЕНИЯ РАСПИСАНИЯ СЕССИИ

Глава посвещена обзору алгоритмов, которые можно применять для составления расписания сессии, а также для оптимизации этого процесса.

В параграфе 2.1 приведена постановка задачи составления расписания сессии, далее в параграфе 2.2 рассморены два возможных режима составления расписания, а в параграфе 2.3 - алгоритмы, используемые для решения данной задачи.

2.1. Постановка задачи составления расписания сессии

Для составления расписания учебной сессии в университете необходимо каждой запланированной аттестации подобрать день, время и аудиторию проведения. Известны доступные аудитории и множество аттестаций, которые необходимо провести в рамках сессии. Для каждой аудитории известно время ее доступности. Для каждого типа аттестации определена длительность, максимальное количество в день и количество дней отдыха до и после её проведения. Нужно составить расписание сессии так, чтобы общая эффективность расписания была наибольшей.

2.1.1. Обозначения

Составим расписание сесии для g учебных групп, каждая из которых характеризуется номером и количеством учащихся в ней студентов.

Множество номеров учебных групп:

$$Gr = \{gr_1,..,gr_g\} \tag{2.1}$$

Количество студентов в соответсвующей учебной группе:

$$S = \{s_1,..,s_g\} \tag{2.2}$$

В данной задаче необходимо учитывать пожелания преподавателей, поэтому каждый преподаватель помимо ФИО должен характеризоваться множеством удобных для него дней и часов проведения аттестаций. Также преподавателям необходимо прописывать приоритет, чтобы в случаях, когда нельзя удовлетворить пожеланиям всех преподавателей, в первую очередь будут учитываться пожеланиям именно преподавателей с большим приоритетом.

Множество, состоящее из p имён преподавателей ВУЗа:

$$T = \{t_1,..,t_p\} \tag{2.3}$$

Приоритеты соответсвующих преподавателей:

$$Pr = \{pr_1,..,pr_p\} \tag{2.4}$$

Множество из q дней, которые преподаватель t выбрал возможными для проведения им аттестаций:

$$Td_t = \{td_{t1},..,td_{tq}\} \tag{2.5}$$

Множество из c часов, которые преподаватель t выбрал возможными для проведения им аттестаций:

$$Tt_t = \{tt_{t1},..,tt_{tc}\} \tag{2.6}$$

Важно также учитывать правила проведения сессий, ограничивающие количество аттестаций в день и определяющие, сколько дней отдыха нужно оставить группе до и после аттестации.

Определим множество из y типов аттестаций (экзамен, зачёт и т.п.):

$$A = \{a_1,..,a_y\} \tag{2.7}$$

Количество дней отдыха перед (Pb) и после (Pa) проведением аттестации каждого типа:

$$Pb = \{pb_1,..,pb_y\} \tag{2.8}$$

$$Pa = \{pa_1,..,pa_y\} \tag{2.9}$$

Длительность каждого типа аттестации в часах:

$$Ad = \{ad_1,..,ad_y\} \tag{2.10}$$

Максимальное количество аттестаций каждого типа в день:

$$Ac = \{ac_1,..,ac_y\} \tag{2.11}$$

При выборе аудиторий для проведения аттестаций необходимо полагаться на их размер и техническую оснащённость. Опрелелим множество из u номеров аудиторий:

$$R = \{r_1,..,r_u\} \tag{2.12}$$

Типы соответсвующих аудиторий (компьютерный класс, имеет проектор и т.п.):

$$Rt = \{rt_1,..,rt_y\} \tag{2.13}$$

Количество мест в соответсвующих аудиториях:

$$Rs = \{rs_1,..,rs_y\} \tag{2.14}$$

Аттестации, которые необходимо провести в течение сессии описываются учбной группой, дисциплиной и преподавателями, которые должны провести данную аттестацию. Так же для каждой аттестации необходимо указать её тип и тип аудитории, в которой она должна проводиться.

Множество аттестаций:

$$E = \{e_1,..,e_k\} \tag{2.15}$$

Множества групп (Eg) и дисциплин (Ec) для каждой из k аттестаций:

$$Eg = \{eg_i \in Gr, \forall i \in \{1,..,k\}\} \tag{2.16}$$

$$Ec = \{ec_1,..,ec_k\} \tag{2.17}$$

Множества типов аттестаций и типов аудиторий для каждой из k аттестаций:

$$Ea = \{ea_i \in A, \forall i \in \{1,..,k\}\} \tag{2.18}$$

$$Er = \{er_1,..,er_k\} \tag{2.19}$$

Множество, состоящее из преподавателей, проводящих аттестацию e:

$$Et_e = \{et_{ei} \in T, \forall i\} \tag{2.20}$$

Каждой аттестации необходимо сопоставить дату, часы и аудиторию для проведения. Декартово произведение всех возможных дат, часов и свободных кабинетов представляет собой множество потенциальных окон для проведения аттестаций.

Множество окон:

$$W = \{w_1,..,w_n\} \tag{2.21}$$

$wd_i \in D$ - календарный день, соответствующий окну i;

$wh_i \in H$ - время, соответствующее окну i;

$wc_i \in R, \forall i \in \{1,..,n\}$ - аудитория, соответствующая окну i.

Введём множество Solutions, состоящее из функций булевых переменных, которые принимают значение 1, если за i-ей аттестацией бронируется j-е окно:

$$S(i,j) = \begin{cases} 1 & \text{the attestation } e_i \text{ is held in window } w_j \\ 0 & \text{the attestation } e_i \text{ is not held in window } w_j \end{cases} \quad (2.22)$$

Таким образом, необходимо найти значения функции $S \in Solutions$ для $\forall i \in \{1,..,k\}$ и $\forall j \in \{1,..,n\}$. Множество пар <i,j>, для которых S(i,j) = 1, представляют собой расписание сессии - соотвествие аттестации дню, врмени и аудитории.

2.1.2. Ограничения

Задача составления расписания сессии имеет ряд физических ограничений, а также ограничений, обусловленных правилами проведения сессии. Соблюдение этих ограничений позволит составить корректное расписание сессии:

Во-первых, в одной аудитории в одно время может проводиться только одна аттестация:

$$\forall j \in \{1,..,n\} \sum_{i=1}^{k} S(i,j) = 1 \quad (2.23)$$

Каждая аттестация в период сессии должна проводиться единожды, так как в данном случае не рассматривается расписание дополнительной сессии:

$$\forall i \in \{1,..,k\} \sum_{j=1}^{n} S(i,j) = Ad_e a_i \quad (2.24)$$

Преподаватель не должен отрабатывать в определённый день больше некоторого количества часов x:

$$\forall b \in \{1,..,p\} : \sum_{\forall i \in \{1,..,k\}, et_{ia}=t_b} \sum_{\forall j \in \{1,..,n\}} \sum_{\forall u \in d_u \in D, wd_j=d_u} S(i,j) <= x \quad (2.25)$$

Любая группа не может сдавать аттестации любого типа в день больше, чем макисмальное число, обусловленное типом аттестации:

$$\forall b \in \{1,..,g\}, \forall j \in \{1,..,n\}, \forall m \in \{1,..,y\} : \sum_{\forall i \in \{1,..,k\}, et_i=a_m, eg_i=gr_b} \sum_{\forall d \in D, wd_j=d} S(i,j) <= ac_m * ad_m \quad (2.26)$$

Любая группа перед любой своей аттестацией не должна иметь аттестаций в окно отдыха перед аттестацией этого типа:

$$\forall b \in \{1,..,g\}, \forall i \in \{1,..,k\} eg_i = gr_b : \max_{\forall j \in \{1,..,n\}, ed_j < ed_i} (ed_j) < ed_i - pb_{et_i} \tag{2.27}$$

Любая группа после любой своей аттестацией не должна иметь аттестаций в окно отдыха после аттестацией этого типа:

$$\forall b \in \{1,..,g\}, \forall i \in \{1,..,k\} eg_i = gr_b : \min \forall j \in \{1,..,n\}, ed_j > ed_i (ed_j) > ed_i - pa_{et_i} \tag{2.28}$$

Каждая аттестация должна проводиться в аудитории равной и или превосходящей по вместимости размеру группы:

$$\forall i \in \{1,...,k\}, \forall j \in \{1,..,n\} : s_{eg_i} > rs_{wc_j} \rightarrow S(i,j) = 0 \tag{2.29}$$

Каждая аттестация не может проводиться в аудитории с оснащённостью меньшей ожидаемой:

$$\forall i \in \{1,...,k\}, \forall j \in \{1,..,n\} : er_i < rt_{wc_j} \rightarrow S(i,j) = 0 \tag{2.30}$$

Чтобы составить расписание, комфортное для преподавательского состава, учтём некоторые ограничения связанные с предпочтениями преподавателей. Эти огнаричения уже не столь критичны, как перечисленные выше, так как их нарушение не влечёт за собой нарушение правил или законов физики, но их наличие делает расписание более удобным для сотрудников.

Каждый преподаватель волен указать список календарных дней, в которые он готов принимать аттестации. Идеальное расписание предполагает, что любой преподаватель проводит все свои аттестации только в дни из этого списка:

$$\forall t \in T, \forall d \in D : d \notin Td_t \rightarrow S(i,j) = 0 \tag{2.31}$$

Также, преподаватель может указать удобные для него часы провеления аттестаций. Идеальное расписнаие учитывает это и располагает аттестации так, чтобы они затрагивали только выбранные часы каждого преподавателя:

$$\forall t \in T, \forall h \in H : h \notin Tt_h \rightarrow S(i,j) = 0 \tag{2.32}$$

Данная задача относится к классу NP-полных задач, а значит в худшем исходе придётся перебрать все возможные k аттестаций и n окон в качестве аргументов

функции S(i,j), где выполняются все описанные выше ограничения. Обобщим их для функции S:

$$S(i,j) = \begin{cases} \forall j \in \{1,..,n\} \sum_{i=1}^{k} S(i,j) = 1; \\ \forall i \in \{1,..,k\} \sum_{j=1}^{n} S(i,j) = Ad_e a_i; \\ \forall b \in \{1,..,p\} : \\ \sum_{\forall i \in \{1,..,k\}, et_{ia}=t_b} \sum_{\forall j \in \{1,..,n\}} \sum_{\forall u \in d_u \in D, wd_j=d_u} S(i,j) <= x; \\ \forall b \in \{1,..,g\}, \forall j \in \{1,..,n\}, \forall m \in \{1,..,y\} : \\ \sum_{\forall i \in \{1,..,k\}, et_i=a_m, eg_i=gr_b} \sum_{\forall d \in D, wd_j=d} S(i,j) <= ac_m * ad_m; \\ \forall b \in \{1,..,g\}, \forall i \in \{1,..,k\} eg_i = gr_b : \\ \max\limits_{\forall j \in \{1,..,n\}, ed_j < ed_i} (ed_j) < ed_i - pb_{et_i}; \\ \forall b \in \{1,..,g\}, \forall i \in \{1,..,k\} eg_i = gr_b : \\ \min\limits_{\forall j \in \{1,..,n\}, ed_j > ed_i} (ed_j) > ed_i - pa_{et_i}; \\ \forall i \in \{1,...,k\}, \forall j \in \{1,..,n\} : s_{eg_i} > rs_{wc_j} \rightarrow S(i,j) = 0; \\ \forall i \in \{1,...,k\}, \forall j \in \{1,..,n\} : er_i < rt_{wc_j} \rightarrow S(i,j) = 0 \end{cases} \tag{2.33}$$

Оптимальным расписанием будет такое, где минимальное количество пожеланий преподавателей игнорируется. Таким образом, сведём задачу к поиску такого расписания S, на котором выполняется слудеющий минимум:

$$\min_{\forall S \in Solutions} \Big(\sum_{\forall i \in \{1,...,k\}, \forall j \in \{1,..,n\}, wd_j \notin Td_{et_i}, wh_j \notin Th_{et_i}} (S(i,j) * pr_{et_i}) \Big) \tag{2.34}$$

Видно, что идеальными расписаниями будут те, где учитываются пожелания всех преподавателей, так как в этом случае точно достигается наименьшее значение минимума формулы (2.34) = 0.

Для решения данной задачи рассмотрим алгоритмы динамического и линейного программирования, алгоритм на графах, эволюционный алгоритм и метод численной оптимизации.

2.2. Режимы алгоритмов составления расписния

Алгоритмы составления расписания можно использовать в двух режимах: инкрементальном и пакетном (batch-режим). В данном разделе рассмотрим, что из себя представляет каждый из этих режимов и почему пакетный режим больше подходит для решения задачи составления расписания сессии.

2.2.1. Инкрементальный режим

Инкрементальным режимом назовём выполнение алгоритма многократно при появлении новых данных. В контексте данной задачи это означает, что каждый раз, когда новый преподаватель будет добавлять информацию о своих предпочтениях, алгоритм будет просчитывать возможные варианты расписания с учётом:

– уже имеющихся данных, которые имеют приоритет, перед новыми;
– внесённых преподавателем изменений.

Так, при каждом новом пересчёте уже составленные расписания преподавателей считаются утверждёнными, что даёт возможность не считать их заново, тем самым ускорив работу алгоритма.

Из преимуществ данного подхода можно выделить возможность преподавателя, не дожидаясь других, увидеть своё расписание. Также, с точки зрения организации, плюсом можно считать стимул преподавателей как можно раньше заполнить форму сбора информации, чтобы иметь приоритет перед другими. Таким образом появляется возможность раньше закончить процесс составление расписания.

Недостатки данного режима более существенны на практике, чем его преимущества, так как основным недостатком является закрепление наиболее удобного расписания за теми, кто заполнил форму сбора раньше остальных, что может повлечь коллизии. Можно рассмотреть случай, когда первый преподаватель, заполнивший форму сбора, выбирает большое окно для проведения экзаменов и алгоритм бронирует за ним несколько определённых случайных дат, а остальные оставляет свободными. Тогда второй преподаватель, уезжающий на конференцию в свободные даты и готовый провести экзамены в даты, занятые первым, уже не сможет это сделать, потому что у первого преподавателя приоритет, и его уже нельзя подвинуть на сводные даты.

2.2.2. Пакетный режим

Пакетным режимом в данном случае будет единоразовая обработка всех полученных сведений. Для этого необходимо определить дедлайн для сбора пожеланий преподаватель, при наступлении которого составить предварительное расписание для всех за один вызов алгоритма.

Преимуществом такого подхода является возможность установить приоритеты преподавателей и групп, на основе которых можно решать коллизии в случае невозможности нахождения идеального решения, учитывающего пожелания каждого.

Недостатком такого подхода можно назвать невозможность до дедлайна получить расписание, но в реальности это не будет глобальным минусом, потому что составленное таким образом предварительное расписание, всё равно, в последствии может быть скорректировано вручную.

Таким образом, из двух рассмотренных режимов для решения задачи составления расписания больше подходит именно пакетный, потому что его преимущества более значимы, а единственный недостаток не играет роли на практике, в отличие от описанных недостатков инкремениального режима.

2.3. Обзор алгоритмов составления расписания сессии

2.3.1. Обход графа в глубину для поиска идеальных решений

Как упоминалось ранее, задача составления расписания сессии является NP-полной. Задача обхода графа тоже NP-полна, и существует множество классических алгоритмов, решающих её. Алгоритм обхода графа в глубину является одним из них. Но чтобы применить этот алгоритм для составления расписания, необходимо свести эту задачу к задаче построения графа, у которого в качестве вершин выступают элементы расписания - аттестации и временные окна для их проведния. Наличие ребра обуславливается выполнением двух следующих условий:

- ребро из вершины с аттестацией i из отсортрованного списка аттестаций E, может быть соединено только с аттестацией i+1 из E;

– ребро из вершины <i,j> с аттестацией i и окном j в вершину <i+1, q> может существовать, если выполняются условия формул (2.33), (2.31) и (2.32) для S(i+1, q) при n=i+1.

Тогда решением будет являться связный граф состоящий из k вершин - пар аттестаций и временных окон, отведённых для этих аттестаций.

Основная идея обхода в глубину – когда возможные пути по ребрам, выходящим из вершин, разветвляются, нужно сначала полностью исследовать одну ветку и только потом переходить к другим веткам. Сложность классического обхода в глубину с матричным заданием графа равна $O(m^2)$, где $m = n * k$ - количество вершин. Но так как известно, что доступность рёбер для каждой из вершин с аттестацией i огрничена n вершинами с аттестацией i+1, сложность снизится до $O(n^2 * k)$. Но так как для проверки остальных условий необходимо проверять i предыдущих вершин графа, сложность останется $O(n^2 * k * \log(k))$.

В процессе решения занчениями функции S(i,j) будет заполняться булева матрица. Важно помнить, что значение 1 в ячейке может быть проставлено, только при соблюдении ограничений системы (2.33) , и стоит заметить, что выполнение ограничения, что в одной аудитории в одно время может проводиться только одна аттестация, влечёт за собой наличие в одном столбце матрицы не более одного значения 1, из-за чего матрицу можно заменить на целочисленный массив, где индексы соответствуют индексу возможного окна, а значения - индексу события, которое будет там проведено.

Будем считать, что идеальное решение найдено, если в каждой строке матрицы есть значение 1 или каждое число от 0 до k встречается в массиве решения, что означает, что каждую аттестацию можно сопоставить какому-то окну w_j с учётом перечисленных выше ограничений. Такое решение добавляется в список всех идеальных решений. Даллее представлен псевдокод процедуры FindSolutions 2.1. , которая реализует алгоритм обхода графа в глубину. Заметим, что в списке доступных окон W все элементы отсортированы по дням, кабинетам и часам, чтобы легко можно было оценивать занятость одного кабинета несколько часов подряд.

В процедуре FindNextStartTime 2.4 проводится проверка условий из формулы (2.33), чтобы подобрать следующее подходящее ребро, исходящее из указанной вершины. В случае, если для вершины не существует исходящего ребра, которое удовлетворяет всем условиям, данная процедура вернёт значение -1, в противном случае вернёт следующее ребро. В данной функции также происходит проверка

```
Algorithm FindSolutions
    Input: 𝔼, 𝕎, k = |𝔼|, n = |𝕎|
    Output: $Solutions
1.  Solutions ← {}, S ← [], i ← 0
2.  while i < k do
3.      time = NextTime(i)
4.      for ∀u ∈ 0,...,k do
5.          if S[u] = i then
6.              S[u] ← −1
7.      nextTime = FindNextStartTime(i,time)
8.      if nextTime = −1 then
9.          if i = 0 then
10.             return Solutions
11.         else
12.             i ← i − 1
13.     else
14.         duration = ed_i
15.         for ∀b ∈ {nextTime,...,nextTime + duration} do
16.             S[b] ← i
17.         i ← i + 1
18.     if i = k then
19.         Solutions ← S
20.         i ← i − 1
21. return Solutions
```

Рис.2.1. Псевдокод алгоритма DFS для составления расписания

условий из формул (2.31) и (2.32), чтобы обеспечить поиск идеального решения, удовлетворяющего пожланиям всех преподавателей.

2.3.2. Алгоритм обхода графа в глубину с учётом приоритетов преподавателей

На практике возникают ситуации, когда недостаток аудиторий и накладки в личных расписаниях преподавателей не позволяют найти такое расписание, которое удовлетворит всем пожеланиям. По этой причине приходится учитывать приоритеты ограничений и минимизировать потери по формуле (2.34).

Существует ряд ограничений, поступиться с которыми нельзя. В их число входят правила проведения аттестаций и физические условия, связанные с проведением экзаменов в аудиториях, опписанные в формуле (2.33). Но есть и менее жёсткие факторы - это пожелания преподавателей к датам и времени экзаменов.

Каждый преподаватель указывает дни и время, когда ему было бы удобно присутствовать на аттестации. В некоторых случаях удобство эквивалентно тому, что в остальные дни преподаватель в принципе не может принимать экзамены. Приоритет таких ограничений должен быть выше, чем у случаев, когда преподавателю просто комфортнее было бы присутствовать на экзаменах в определённые дни.

Для этого было введено множество приоритетов Pr, где больший приоритет соответствует большему влиянию учёта пожеланий преподавателя на потери функции (2.34). Система приоритетов в случае появления коллизий при составлении расписания позволит в первую очередь пытаться нарушить только пожелания с наименьшим приоритетом и только, если это необходимо, с более высокими.

Модернизируем алгоритм поиска «идеального» решения:

Для этого нужно хранить массив ignoreWishes, в котором будут фиксироваться те аттестации, для которых приходится искать решения без учёта предпочтений преподавателей.

Также, в алгоритме 2.3 на строке 2 необходимо отсортировать массив аттестаций по приоритетам преподавателей, чтобы искать решения без учёта высокоприоритетных преподавателей лишь в последнюю очередь.

Метод поиска решений FindSolutions 2.3 теперь содержит условный оператор, который в случае определения того, что идеального решения найти не удаётся, переходит в режим игнорирования пожеланий преподавателя для кон-

Function FindNextStartTime

Input: $event, time, \mathbb{E}, \mathbb{W}, \mathbb{S}, k = |\mathbb{E}|, n = |\mathbb{W}|$

Output: t

1. **if** $time < 0$ **then**
2. **return**-1
3. **for** $i \in \{time,...,k\}$ **do**
4. $w \leftarrow W[i]$
5. **if** $S[i] = -1 \wedge rt_w = er_{event} \wedge i + ad_{event} < n \wedge s_{eg_{event}} <= er_w \wedge er_w = er_{W[i+ad_{event}]} \wedge ed_w = ed_{W[i+ad_{event}]} \wedge wd \in Td_{event} \wedge wt \in Tt_{event} \wedge wt + ed_w \in Tt_{event}$ **then**
6. $teacherTime \leftarrow 0, countPerDay \leftarrow 0, j \leftarrow 0$
7. **while** $i < k$ **do**
8. **if** $S[j] = -1$ **then**
9. $j \leftarrow j + 1$
10. $continue$
11. **if** $w = S[j] \wedge wt \in et_{solution[j]}$ **then**
12. $break$
13. **if** $et_{S[j]} = et_{event}$ **then**
14. $teacherTime = teacherTime + 1$
15. **if** $teacherTime > maxTimePerDay - ed_{event}$ **then**
16. $break$
17. **if** $eg_{S[j]} = eg_{event}$ **then**
18. **if** $(ea_{S[j]=ea_{event}}) \wedge (wd = ed_{S[j]})$ **then**
19. $countPerDay \leftarrow countPerDay + 1$
20. **if** $(ac_{et_{event}} < countPerDay) \vee (wt - pb_{ea_{event}} <= wd_{S[j]} \wedge wt + pa_{ea_{event}} >= wd_{S[j]})$ **then**
21. $break$
22. $k \leftarrow k + 1$
23. **if** $j = n$ **then**
24. **return** i
25. **return**-1

Рис.2.2. Проверка условий формул (2.33), (2.31) и (2.32)

кретного события. Для этого в массиве ignoreWishes выставляется true по индексу, соотвествующему этому событию.

Algorithm FindSolutions

Input: $\mathbb{P}, \mathbb{E}, \mathbb{W}, k = |\mathbb{E}|, n = |\mathbb{W}|$

Output: $\mathbb{S}$

1. $ignoreWishes \leftarrow [], S \leftarrow [], i \leftarrow 0$
2. $Sort(E)$
3. **while** $i < k$ **do**
4. $\quad time = NextTime(i)$
5. $\quad$ **for** $\forall u \in 0,...,k$ **do**
6. $\quad\quad$ **if** $S[u] = i$ **then**
7. $\quad\quad\quad S[u] \leftarrow -1$
8. $\quad nextTime = FindNextStartTime(i,time)$
9. $\quad$ **if** $ignoreWishes[i] = False \wedge nextTime = -1$ **then**
10. $\quad\quad ignoreWishes[i] = True$
11. $\quad\quad nextTime = FindNextStartTime(i,0)$
12. $\quad$ **if** $nextTime = -1$ **then**
13. $\quad\quad$ **if** $i = 0$ **then**
14. $\quad\quad\quad$ **return** $null$
15. $\quad\quad$ **else**
16. $\quad\quad\quad ignoreWishes[i] = False$
17. $\quad\quad\quad i \leftarrow i - 1$
18. $\quad$ **else**
19. $\quad\quad duration = ed_i$
20. $\quad\quad$ **for** $\forall b \in \{nextTime,...,nextTime + duration\}$ **do**
21. $\quad\quad\quad S[b] \leftarrow i$
22. $\quad\quad i \leftarrow i + 1$
23. **return** S

Рис.2.3. Псевдокод алгоритма DFS для составления расписания с учётм приоритетов преподавателей

Метод поиска следующего подходящего времени и аудитории теперь может работать в двух режимах. В режими игнорирования рассматриваются только те окна, которые хотя бы в какой-то мере не удовлетворяют пожеланиям преподавателя. Для этого в алгоритме 2.4 в строке 5 используется логический оператор XOR между результатом проверки соблюдения пожеланий преподавателя и значением в массиве игнорирования.

Function FindNextStartTime

Input: $event, time, \mathbb{E}, \mathbb{W}, \mathbb{S}, k = |\mathbb{E}|, n = |\mathbb{W}|, ignoreWishes$

Output: t

1. **if** $time < 0$ **then**
2. **return**-1
3. **for** $i \in \{time,...,k\}$ **do**
4. $w \leftarrow W[i]$
5. $teacherWish \leftarrow (wd \in Td_{event} \wedge wt \in Tt_{event} \wedge wt + ed_w \in Tt_{event}) \nrightarrow ignoreWishes[i]$
6. **if** $S[i] = -1 \wedge rt_w = er_{event} \wedge i + ad_{event} < n \wedge s_{eg_{event}} <= er_w \wedge er_w = er_{W[i+ad_{event}]} \wedge ed_w = ed_{W[i+ad_{event}]} \wedge teacherWish$ **then**
7. ...
8. **if** $j = n$ **then**
9. **return** i
10. **return**-1

Рис.2.4. Проверка условий формулы (2.33) с учётом режима игнорирования

Асимптотическая сложность этой интерпретации алгоритма тоже равна $O(n^2 * k)$, но требует дополнительную память на хранение бинарного массива ignoreWishes.

2.4. Методы оптимизации поиска лучших решений

2.4.1. Метрики оценки качества расписания

Для оценки предложенных расписаний подсчитаем несколько метрик:

Длительность сессии для каждого преподавателя с учётом его приоритета:

$$DurationT = \sum_{\forall t \in T} (pr_t * (\max_{i \in \{0,\ldots,n\}, et_{S[i]}=t} (wd_i) - \min_{i \in \{0,\ldots,n\}, et_{S[i]}=t} (wd_i))) \quad (2.35)$$

Считаем, что расписание преподавателя должно быть максимально компактным по датам, а значит разницу между первым и последним днём проведения экзаменов нужно минимизировать.

Суммарная длительность минимального отдыха между экзаменами по группам:

$$Pause = \sum_{\forall gr \in Gr} (\min_{i,j \in \{0,\ldots,n\}, eg_{S[i]}=gr, wd_i \neq wd_j, i>j} (wd_i - wd_j)) \quad (2.36)$$

Считаем, что студентам желательно не иметь маленьких перерывов между экзаменами, поэтому стремимся максимизировать Pause для расписания.

Суммарная длительность сессии по группам:

$$DurationG = \sum_{\forall gr \in Gr} \max_{i \in \{0,\ldots,n\}, eg_{S[i]}=gr} (wd_i j) \quad (2.37)$$

Чем раньше закончится сессия, тем раньше у иногородних студентов появится возможность уехать домой, а значит порядковый номер последнего дня сессии для группы нужно пытаться минимизировать.

Суммарное кол-во рабочих дней для преподавателей с учётом их приоритетов:

$$Work_t = \{wd_i, \forall i \; in 1,\ldots,n, et_{S[i]} = t\}; \quad (2.38)$$

$$WDays = \sum_{\forall t \in T} (pr_t * \|Work_t\|) \quad (2.39)$$

Считаем, что преподавателю удобно иметь максимальное количество дней, свободных от проведения экзаменов, а значит метрику WDays стараемся минимизировать. На практике это означает, что лучше провести несколько зачётов в один день, чем заставлять человека несколько раз в неделю приезжать в университет.

Метод getRates(int[] sol) позволит рассчитать все описанные выше метрики для конкретного расписания и вернёт их в качестве одномерного массива длины 4.

2.4.2. Метод ветвей и границ

Метод ветвей и границ принадлежит к группе алгоритмов целочисленного линейного программирования и применяется для оптимизации задач полного

перебора. Он как нельзя кстати придётся для модернизации алгоритма полного обхода графа в глубину. Суть этого метода состоит в том, чтобы в процессе обхода графа, отбрасывать заведомо менее оптимальные решения, чем у же найденные.

Такой подход потребует выделить память под хранение метрик качества. В памяти необходимо держать качество текущего расчитываемого расписания и предыдущего лучшего расписания. В качестве метрики качества возьмём разброс между первым и последним днём проведения сессии для каждого преподавателя. Чем он меньше, тем лучше считается расписание. Естественно, можно выбрать и другие критерии качества расписания, но алгоритм при этом поменяется незначительно. Выбор такой метрики вынуждает хранить массив с количеством дней разброса для каждого преподавателя, поэтому выделим память под масиивы metrics[] и bestMetrics[] размером p (число преподавателей).

Модернизируем функцию поиска решений методом обхода графа в глубину, добавив в качестве условия перехода к следующей аттестации проверку на то, является ли расписание на данном этапе, худшим, чем предыдущее полностью составленное расписание. Эта проверка добавляется в строке 18 алгоритма 2.5.

Функция isItWorse 2.6 пересчитывает для рассматриваемого преподавателя длительность его сессии и проверяет, не является ли оно худшим, чем у того же преподавателя в предыдущем полном расписании. Для этого в массиве текущего решения находится максимальный и минимальный день, для рассматриваемого преподавателя и возвращается их разница.

2.4.3. Алгоритм иммитации отжига

Для выбора лучшего из расписаний и расчёта всех метрик необходимо будет дополнительно обойти x расписаний, состоящих из n возможных временных окон. А потом просуммировать их по p преподавателям и z учебным группам. Если входные данные были такими, что было найдено 5-10 возможных расписаний, то не составит труда просчитать их все, так как относительно n (временные окна по аудиториям), оно вносит минимальный вклад в сложность вычислений.

Но рассмотрим случай, когда методом полного перебора было найдено большое количество возможных расписаний, и число x достаточно велико и даже сравнимо с n. В такой ситуации было бы полезно находить более оптимальные решения без расчёта метрик для каждого расписания, жертвуя некоторой погрешностью.

Algorithm FindSolutions

Input: $\mathbb{E}$, $\mathbb{W}$, $k = |\mathbb{E}|, n = |\mathbb{W}|$

Output: $\mathbb{S}olutions$

1. $metrics \leftarrow [], bestMetrics \leftarrow [], Solutions \leftarrow \{\}, S \leftarrow [], i \leftarrow 0$
2. **while** $i < k$ **do**
3. $\quad time = NextTime(i)$
4. $\quad$ **for** $\forall u \in 0,...,k$ **do**
5. $\quad\quad$ **if** $S[u] = i$ **then**
6. $\quad\quad\quad S[u] \leftarrow -1$
7. $\quad nextTime = FindNextStartTime(i,time)$
8. $\quad$ **if** $nextTime = -1$ **then**
9. $\quad\quad$ **if** $i = 0$ **then**
10. $\quad\quad\quad$ **return** $Solutions$
11. $\quad\quad$ **else**
12. $\quad\quad\quad i \leftarrow i - 1$
13. $\quad$ **else**
14. $\quad\quad duration = ed_i$
15. $\quad\quad$ **for** $\forall b \in \{nextTime,...,nextTime + duration\}$ **do**
16. $\quad\quad\quad S[b] \leftarrow i$
17. $\quad\quad i \leftarrow i + 1$
18. $\quad\quad$ **if** $i = 0 \vee \neg isItWorse(i)$ **then**
19. $\quad\quad\quad i \leftarrow i + 1$
20. $\quad\quad$ **else**
21. $\quad\quad\quad$ **return** $Solutions$
22. $\quad$ **if** $i = k$ **then**
23. $\quad\quad Solutions \leftarrow S$
24. $\quad\quad bestMetrics \leftarrow metrics$
25. $\quad\quad i \leftarrow i - 1$
26. **return** $Solutions$

Рис.2.5. Псевдокод алгоритма DFS с использованием метода ветвей и границ

Function isItWorse

Input: $event, time, \mathbb{E}, \mathbb{W}, \mathbb{S}, k = |\mathbb{E}|, n = |\mathbb{W}|, ignoreWishes$

Output: t

1. max,$min \leftarrow wd_{n-1}, i \leftarrow event - 1$
2. **while** $i >= 0 \wedge et_i = et_{event}$ **do**
3. $\quad j \leftarrow \min_{j \in 0,k-1,S[j]=i}(j)$
4. $\quad$ **if** $wd_j > max$ **then**
5. $\quad\quad max = wd_j$
6. $\quad$ **else if** $wd_j < min$ **then**
7. $\quad\quad min = wd_j$
8. $\quad i \leftarrow i - 1$
9. $metrics[event] \leftarrow max - min$
10. **return** $bestMetrics[0] \neq -1 \wedge bestMetrics[et_{event}] < metrics[et_{event}]$

Рис.2.6. Функция сравнения метрики качества для преподавателя с предыдущим лучшим решением

Для этого можно применить оптимизационный алгоритм имитации отжига. Он основан на имитации физического процесса отжига металлов - при постепенно понижающейся температуре переход атома из одной ячейки кристаллической решётки в другую происходит с некоторой вероятностью, которая понижается с понижением температуры.

При помощи моделирования такого процесса ищется такая точка или множество точек, на котором достигается минимум некоторой числовой функции F(x), где x принадлежит множеству возможных решений.

Для задачи выбора лучшего расписания функция F(x) - функция, вычисляющая качество расписания, например, по метрике S (суммарная продолжительность сессии для каждой группы) из раздела Метрики выбора лучшего решения, которая рассчитывается в методе getRate(int i), где i - индекс расписания в массиве решений.

ГЛАВА 3. НАЗВАНИЕ ТРЕТЬЕЙ ГЛАВЫ: РАЗРАБОТКА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Хорошим стилем является наличие введения к главе. Во введении может быть описана цель написания главы, а также приведена краткая структура главы.

3.1. Название параграфа

3.2. Название параграфа

3.3. Выводы

Текст выводов по главе 3.

3.4. Название параграфа

Название параграфа оформляется с помощью команды `\section{...}`, название главы — `\chapter{...}`.

ГЛАВА 4. НАЗВАНИЕ ЧЕТВЁРТОЙ ГЛАВЫ. АПРОБАЦИЯ РЕЗУЛЬТАТОВ ИССЛЕДОВАНИЯ, А ИМЕННО: МЕТОДА, АЛГОРИТМА, МОДЕЛИ ИССЛЕДОВАНИЯ

Хорошим стилем является наличие введения к главе. Во введении может быть описана цель написания главы, а также приведена краткая структура главы.

4.1. Название параграфа

4.2. Название параграфа

Пример ссылки на литературу [**avtonomova:fya**; **Peskov2004-ru**; **Kotelnikov2004-ru**; **Kotelnikov2004**].

4.3. Выводы

Текст выводов по главе 4.

ЗАКЛЮЧЕНИЕ

Заключение (2 – 5 страниц) обязательно содержит выводы по теме работы, *конкретные предложения и рекомендации* по исследуемым вопросам. Количество общих выводов должно вытекать из количества задач, сформулированных во введении выпускной квалификационной работы.

Предложения и рекомендации должны быть органически увязаны с выводами и направлены на улучшение функционирования исследуемого объекта. При разработке предложений и рекомендаций обращается внимание на их обоснованность, реальность и практическую приемлемость.

Заключение не должно содержать новой информации, положений, выводов и т. д., которые до этого не рассматривались в выпускной квалификационной работе. Рекомендуется писать заключение в виде тезисов.

Последним абзацем в заключении можно выразить благодарность всем людям, которые помогали автору в написании ВКР.

СПИСОК СОКРАЩЕНИЙ И УСЛОВНЫХ ОБОЗНАЧЕНИЙ

DOI Digital Object Identifier.

WoS Web of Science.

ВКР Выпускная квалификационная работа.

ТГ-объект Текстово-графический объект.

СЛОВАРЬ ТЕРМИНОВ

TeX — язык вёрстки текста и издательская система, разработанные Дональдом Кнутом.

LaTeX — язык вёрстки текста и издательская система, разработанные Лэсли Лампортом как надстройка над TeX.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

Приложение 1

Краткие инструкции по настройке издательской системы LaTeX

В SPbPU-BCI-template *автоматически выставляются необходимые настройки и в исходном тексте шаблона приведены примеры оформления текстово-графических объектов*, поэтому авторам достаточно заполнить имеющийся шаблон текстом главы (статьи), не вдаваясь в детали оформления, описанные далее. Возможный «быстрый старт» оформления главы (статьи) под Windows следующий[П1.1]:

A. Установка полной версии MikTeX [**latex-miktex**]. В процессе установки лучше выставить параметр доустановки пакетов «на лету».
B. Установка TexStudio [**latex-texstudio**].
C. Запуск TexStudio и компиляция `my_chapter.tex` с помощью команды «Build&View» (например, с помощью двойной зелёной стрелки в верхней панели). *Иногда, для достижения нужного результата необходимо несколько раз скомпилировать документ.*
D. В случае, если не отобразилась библиография, можно
   - воспользоваться командой Tools → Commands → Biber, затем запустив Build&View;
   - настроить автоматическое включение библиографии в настройках Options → Configure TexStudio → Build → Build&View (оставить по умолчанию, если сборка происходит слишком долго): `txs:///pdflatex | txs:///biber | txs:///pdflatex | txs:///pdflatex | txs:///view-pdf`.

В случае возникновения ошибок, попробуйте скомпилировать документ до последних действий или внимательно ознакомьтесь с описанием проблемы в log-файле. Бывает полезным переход (по подсказке TexStudio) в нужную строку в pdf-файле или запрос с текстом ошибке в поисковиках. Наиболее вероятной проблемой при первой компиляции может быть отсутствие какого-либо установленного пакета LaTeX.

В случае корректной работы настройки «установка на лету» все дополнительные пакеты будут скачиваться и устанавливаться в автоматическом режиме. Если доустановка пакетов осуществляется медленно (несколько пакетов за один запуск

[П1.1] Внимание! Пример оформления подстрочной ссылки (сноски).

компилятора), то можно попробовать установить их в ручном режиме следующим образом:

1. Запустите программу: меню → все программы → MikTeX → Maintenance (Admin) → MiKTeX Package Manager (Admin).
2. Пользуясь поиском, убедитесь, что нужный пакет присутствует, но не установлен (если пакет отсутствует воспользуйтесь сначала MiKTeX Update (Admin)).
3. Выделив строку с пакетом (возможно выбрать несколько или вообще все неустановленные пакеты), выполните установку Tools → Install или с помощью контекстного меню.
4. После завершения установки запустите программу MiKTeX Settings (Admin).
5. Обновите базу данных имен файлов Refresh FNDB.

Для проверки текста статьи на русском языке полезно также воспользоваться настройками Options → Configure TexStudio → Language Checking → Default Language. Если русский язык «ru_RU» не будет доступен в меню выбора, то необходимо вначале выполнить Import Dictionary, скачав из интернета любой русскоязычный словарь.

Далее приведены формулы (П1.2), (П1.1), рис.П1.2, рис.П1.1, табл.П1.2, табл.П1.1.

$$\pi \approx 3{,}141. \tag{П1.1}$$

Рис.П1.1. Вид на гидробашню СПбПУ [**spbpu-gallery**]

Таблица П1.1

Представление данных для сквозного примера по ВКР [**Peskov2004**]

G	m_1	m_2	m_3	m_4	K
g_1	0	1	1	0	1
g_2	1	2	0	1	1
g_3	0	1	0	1	1
g_4	1	2	1	0	2
g_5	1	1	0	1	2
g_6	1	1	1	2	2

П1.1. Параграф приложения

П1.1.1. Название подпараграфа

Название подпараграфа оформляется с помощью команды `\subsection{...}`. Использование подподпараграфов в основной части крайне не рекомендуется.

П1.1.1.1. Название подподпараграфа

$$\pi \approx 3{,}141. \tag{П1.2}$$

Рис.П1.2. Вид на гидробашню СПбПУ [**spbpu-gallery**]

Таблица П1.2

Представление данных для сквозного примера по ВКР [**Peskov2004**]

G	m_1	m_2	m_3	m_4	K
g_1	0	1	1	0	1
g_2	1	2	0	1	1
g_3	0	1	0	1	1
g_4	1	2	1	0	2
g_5	1	1	0	1	2
g_6	1	1	1	2	2

Приложение 2

Некоторые дополнительные примеры

В приложении[П2.1] приведены формулы (П2.2), (П2.1), рис.П2.2, рис.П2.1, табл.П2.2, табл.П2.1

$$\pi \approx 3{,}141. \qquad \text{(П2.1)}$$

Рис.П2.1. Вид на гидробашню СПбПУ [**spbpu-gallery**]

Таблица П2.1

Представление данных для сквозного примера по ВКР [**Peskov2004**]

G	m_1	m_2	m_3	m_4	K
g_1	0	1	1	0	1
g_2	1	2	0	1	1
g_3	0	1	0	1	1
g_4	1	2	1	0	2
g_5	1	1	0	1	2
g_6	1	1	1	2	2

[П2.1] Внимание! Пример оформления подстрочной ссылки (сноски).

П2.1. Подраздел приложения

$$\pi \approx 3{,}141. \qquad \text{(П2.2)}$$

Рис.П2.2. Вид на гидробашню СПбПУ [**spbpu-gallery**]

Таблица П2.2

Представление данных для сквозного примера по ВКР [**Peskov2004**]

G	m_1	m_2	m_3	m_4	K
g_1	0	1	1	0	1
g_2	1	2	0	1	1
g_3	0	1	0	1	1
g_4	1	2	1	0	2
g_5	1	1	0	1	2
g_6	1	1	1	2	2