Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Воронежский государственный технический университет»
(ФГБОУ ВО ВГТУ)

На правах рукописи

Дятчина Анастасия Владимировна

УПРАВЛЕНИЕ ПРОЦЕССАМИ ПРОЕКТНОЙ ДЕЯТЕЛЬНОСТИ IT-КОМПАНИИ НА ОСНОВЕ ОПТИМИЗАЦИОННЫХ МОДЕЛЕЙ РАСПРЕДЕЛЕНИЯ ИСПОЛНИТЕЛЕЙ И ВРЕМЕННЫХ РЕСУРСОВ

Специальность 2.3.4. Управление в организационных системах

Диссертация
на соискание учёной степени
кандидата технических наук

Научный руководитель:
доктор технических наук, профессор
Олейникова Светлана Александровна

Воронеж-2025

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Актуальность темы. В настоящее время информационные технологии являются неотъемлемой частью любых организационных систем. IT-проекты обеспечивают инфраструктурную и информационную поддержку организаций, способствуют взаимодействию с конечными пользователями и реализуют широкий спектр бизнес-функций. На этом фоне особую актуальность приобретает задача эффективного управления деятельностью IT-компаний как организационных систем. Это комплексная и многоплановая проблема, требующая системного подхода, а также вовлечения команды специалистов, численность и структура которой зависят от масштаба организации. Одним из центральных аспектов управления становится грамотное распределение задач между исполнителями и корректная оценка сроков их выполнения, что, в конечном итоге, позволяет оптимизировать внутренние процессы и повысить финансовую отдачу.

Данная задача частично относится к классам задач управления проектами и задач о назначении. В настоящее время многие проблемы в этих областях достаточно глубоко изучены и получены соответствующие методы. Так, исследованием и развитием методов решения задач в области управления проектами в организационных системах занимались такие ученые, как Новиков Д.А., Бурков В.Н., Гимади Э.Х., Голенко-Гинзбург Д.И. и др. Разработкой методов, позволяющих получить осуществлять управление коллективом организационных систем путем поиска оптимального соответствия между исполнителями и задачами, занимались Г.Кун, Л. Кауфман и др.

Тем не менее, управление организационной деятельностью IT-компании обладает рядом специфических черт, существенно отличающих их от классических проектных моделей. Во-первых, многие задачи требуют регулярного взаимодействия с заказчиком, что может не только изменить сроки исполнения, но и привести к появлению новых задач. Во-вторых,

характерной чертой IT-команд является их неоднородность: сотрудники обладают разным уровнем квалификации, что влияет на скорость, качество и надежность выполнения задач. Кроме того, нестабильность внешних и внутренних факторов делает точное прогнозирование сроков на старте проекта затруднительным. Таким образом, эффективное управление IT-компаниями требует адаптации существующих математических и алгоритмических методов с учетом отраслевой специфики и высокой степени неопределенности.

Целью работы является разработка моделей и алгоритмов управления деятельностью IT-компании как организационной системы путем определения наилучших сроков для отдельных задач IT-проекта, допускающих возможность их повторного выполнения или коррекции, а также назначения им исполнителей в зависимости от их квалификации.

Задачи исследования. Для достижения поставленной цели необходимо решить следующие задачи:

1. Провести анализ существующих подходов к управлению процессами проектной деятельности в IT-компании как организационной системы в рамках поиска наилучшего подхода к управлению коллективом исполнителей, распределению задач и планирования проектов.

2. Разработать модель управления проектом в IT-компании как организационной системе, отличающуюся учетом квалификации специалистов, их занятости, типизации задач и обеспечивающей наилучшую организацию деятельности IT-компании путем планирования времени начала каждой из задач проекта и подбора для нее исполнителя.

3. Разработать математическую модель оценки сроков выполнения как отдельных задач, так и всего IT-проекта данной организационной системы, отличающуюся учетом коррекции задач, обеспечивающую повышенную точность по сравнению с аналогами.

4. Разработать алгоритмы организации работ в IT-компании, отличающиеся одновременным решением задач определения сроков и

времени начала для каждого из заданий IT-проекта и подбора для них наилучших исполнителей и обеспечивающие возможность повышения качества управления деятельностью IT-компании за счет более рационального использования человеческих ресурсов и более точных оценок времени выполнения как всего проекта, так и его отдельных заданий.

5. Реализовать программное обеспечение системы управления IT-компанией как организационной системой, отличающееся интеграцией систем имитационного моделирования, планирования и управления и базы данных, и обеспечивающего оптимизацию деятельности IT-компании с точки зрения выбора оптимальных сроков решения каждой из задач IT-проекта и подбора наилучшего исполнителя для нее.

6. Провести экспериментальное исследование эффективности разработанных моделей и алгоритмов управления IT-компанией как организационной системой с точки зрения длительности IT-проекта.

Объект исследования: деятельность IT-компании как организационной системы, проекты которой допускают высокую степень неопределенности.

Предмет исследования: математические модели и алгоритмы, оптимизирующие процесс управления деятельности в IT-компании с особенностями, указанными выше.

Методы исследования. При решении поставленных в диссертации задач использовались методы управления проектами, теории принятия решений, исследования операций, математического моделирования, теории вероятностей и математической статистики, методы оптимизации, а также методы объектно-ориентированного программирования.

Тематика работы соответствует следующим пунктам паспорта специальности 2.3.4 «Управление в организационных системах»: п.2 «Разработка математических моделей и критериев эффективности, качества и надёжности организационных систем»; п.3 «Разработка методов и алгоритмов решения задач управления в организационных системах.»; п.4 «Разработка

информационного и программного обеспечения систем управления и механизмов принятия решений в организационных системах».

Научная новизна работы. В диссертации получены следующие результаты, характеризующиеся научной новизной:

1. Модель управления проектом в IT-компании как организационной системе, отличающаяся учетом квалификации специалистов, их занятости, типизации задач и обеспечивающая наилучшую организацию деятельности IT-компании путем планирования времени начала каждой из задач проекта и подбора для нее исполнителя.

2. Математическая модель оценки сроков выполнения как отдельных задач, так и всего IT-проекта данной организационной системы, отличающаяся учетом коррекции задач, обеспечивающая повышенную точность по сравнению с аналогами.

3. Алгоритмы организации работ в IT-компании, отличающиеся одновременным решением задач определения сроков и времени начала для каждого из заданий IT-проекта и подбора для них наилучших исполнителей и обеспечивающие возможность повышения качества управления деятельностью IT-компании за счет более рационального использования человеческих ресурсов и более точных оценок времени выполнения как всего проекта, так и его отдельных заданий.

4. Структура программного обеспечения системы управления деятельностью IT-компании как организационной системы, отличающаяся комплексной интеграцией подсистем имитационного моделирования, планирования, оптимизации и базы данных, и обеспечивающая автоматизацию процесса управления IT-компанией путем реализации разработанных алгоритмов.

Практическая значимость исследования заключается в разработке программного обеспечения системы управления, ориентированного на решение задачи оптимизации планирования деятельности IT-компании путем назначения отдельным задачам исполнителей с учетом их различной

квалификации, а также допущения возможной последующей коррекции плана из-за перераспределения задач в случае выбытия исполнителя или внесения корректировок заказчиком.

Положения, выносимые на защиту

1. Модель управления проектом в IT-компании как организационной системе позволяет обеспечить наилучшую организацию деятельности IT-компании путем планирования времени начала каждой из задач проекта и подбора для нее исполнителя

2. Математическая модель оценки сроков выполнения как отдельных задач, так и всего IT-проекта данной организационной системы обеспечивает повышенную точность по сравнению с аналогами.

3. Алгоритмы организации работ в IT-компании обеспечивают возможность повышения качества управления деятельностью IT-компании за счет более рационального использования человеческих ресурсов и более точных оценок времени выполнения как всего проекта, так и его отдельных заданий.

4. Структура программного обеспечения системы управления IT-компанией как организационной системой обеспечивает автоматизацию процесса управления IT-компанией путем реализации разработанных алгоритмов.

Результаты внедрения. Разработанный программный комплекс в составе программного обеспечения, предназначенного для поддержки принятия решений о стимулировании исполнителей и назначении им работ, апробирован в ЗАО «Микрон» (г. Липецк). Разработанные модели и алгоритмы управления проектами внедрены в учебный процесс ФГБОУ ВО «Воронежский государственный технический университет» (дисциплина «Проектная деятельность»).

Апробация работы. Основные положения диссертационной работы докладывались и обсуждались на следующих научно-практических конференциях: IX Международной научно-практической конференции

«Прикладная математика и информатика: современные исследования в области естественных и технических наук» (Тольятти, 2023), XIII Международной молодежной научно-практической конференции «Математическое моделирование процессов и систем» (Стерлитамак, 2023), XVIII международной научно-практической конференции «Антропоцентрические науки в образовании» (Воронеж, 2023), Международной научной конференция «Актуальные проблемы прикладной математики, информатики и механики» (Воронеж, 2024), XXI международной научно-технической конференции «Новые информационные технологии и системы» (Пенза, 2024), международной научно-практической конференции «Интеллектуальные информационные системы» (Воронеж, 2025), открытой Международной научно-практической конференции «Управление программным инжинирингом» (Воронеж, 2025), Информационные технологии в моделировании и управлении: подходы, методы, решения (Тольятти, 2025).

Публикации. По результатам диссертационного исследования опубликовано 18 научных работ, в том числе 4 – в изданиях, рекомендованных ВАК РФ, и получено одно свидетельство о регистрации программы. В работах, опубликованных в соавторстве и приведенных в конце автореферата, лично автором получены следующие результаты:

[34] – модель управления IT-проектом, [33, 70] - математическая модель оценки сроков выполнения отдельных задач IT-проекта, [30, 36, 69, 105] – математическая и имитационная модель оценки сроков IT-проекта, [29, 65, 72] - алгоритм решения задачи организации работ в IT-компании, [28, 31, 35] – реализация системы управления деятельностью IT-компании путем организации работы исполнителей и планирования IT-проекта, [32, 66, 68, 73] – анализ эффективности разработанных алгоритмов решения задач планирования, [71] - структура программного обеспечения системы управления деятельностью IT-компании,

Структура и объем работы. Диссертационная работа состоит из вве-

дения, четырех глав, заключения, списка литературы из 120 наименований. Основная часть работы изложена на 145 страницах, содержит 38 рисунков, 24 таблиц.

ГЛАВА 1. СОВРЕМЕННОЕ СОСТОЯНИЕ ПРОБЛЕМЫ УПРАВЛЕНИЯ ПРОЦЕССАМИ ПРОЕКТНОЙ ДЕЯТЕЛЬНОСТИ IT-КОМПАНИИ

Глава посвящена анализу современного состояния проблемы управления процессами деятельности IT-компании с целью оптимизации ее деятельности. В первой главе выявлены ключевые проблемы управления процессами проектной деятельности IT-компании. Особенности распределения задач по исполнителям и исследование существующих методов решения данной проблемы приведены во второй части главы. Анализ существующих подходов к оценке длительности выполнения отдельных задач и возможности их применения в области управления IT-проектами проанализированы в третьей части. В четвертой части главы сформулирована цель и задачи диссертационного исследования.

1.1 Ключевые проблемы управления процессами проектной деятельности IT-компании

В современном обществе IT-отрасль занимает все больший объем [8, 20], поскольку практически вся деятельность предприятий и организаций полностью или частично переходит на онлайн площадки. Массовый рост IT-компаний возник из-за множества факторов [39, 90], к которым можно отнести как технические (снижение стоимости оборудования, развитие интернет-инфраструктуры, создание доступных (open-source) инструментов обработки и т.д.) [46], так и экономические (рост спроса на онлайн-услуги со стороны населения) [90] и социальные (повышение грамотности населения, смена потребительских привычек и т.д.) [93]. Данные факторы создали рыночную необходимость и стимулы для распространения этой сферы.

В связи с этим, возникла необходимость эффективного управления деятельностью IT-компании [4, 55]. Для решения данной задачи необходимо проанализировать специфику данной деятельности и выявить основные

отличия IT-компании от других организаций. Эта специфика определяется особенностями выпускаемых продуктов (в данном случае, IT-проектов). В первую очередь, к ним относится высокая неопределенность требований. В отличии от другой продукции, при создании IT-проекта возможны существенные изменения требований заказчика, предъявляемые к отдельным составляющим продукта. В ряде случаев это имеет место из-за невозможности точной формулировки заказчиком своих предпочтений. Это приводит к недопониманию между заказчиком и исполнителем и созданию в конечном итоге продукта, не соответствующего ожиданиям заказчика. В других случаях требования заказчика изначально противоречивы и размыты. Они могут меняться под влиянием новых факторов (например, обратной связи от пользователей в ходе опытного тестирования программного продукта, изменения каких-либо условий и т.д.). Кроме того, у заказчика могут возникнуть новые идеи и пожелания в результате демонстрации промежуточных результатов. Все это говорит о том, что для IT-проектов весьма существенным является фактор неопределенности и изменчивости требований, что делает классическое планирование малоэффективным и требует постоянной адаптации.

Второй особенностью IT-проектов является ярко выраженная технологическая динамика. IT-отрасль характеризуется интенсивным появлением новых языков программирования, библиотек, платформ, технологий и других технических средств и инструментов. Это может привести к тому, что в ходе выполнения проекта выбранные технические инструменты и средства могут оказаться неэффективными. В ряде случаев для таких проектов осуществляется пересмотр технологий «на лету», что создает дополнительные риски и сложности. Это фактически единственная отрасль, где IT-команда вынуждена постоянно осваивать новые технологии, проводить исследования и эксперименты по их внедрению, что, очевидно, сказывается на качестве и скорости разработки.

Третья существенная особенность IT-команд заключается в разном уровне ее исполнителей. С учетом крайне высокой (по сравнению с другими отраслями) оплатой труда IT-разработчиков, руководителям компаний выгодно составлять команду из исполнителей разных уровней. Это обусловлено тем, что задачи, на которые декомпозируется IT-проект, бывают разного уровня сложности [58]. Следовательно, наличие в команде лишь высококвалифицированных разработчиков высокого уровня при наличии таких задач будет невыгодно с финансовой точки зрения. Это говорит о том, что производительность разных разработчиков будет принципиально разной. В связи с этим, существующие подходы оценки времени решения отдельной задачи (например, метод PERT), будут малоэффективными [39]. Кроме того, это делает достаточно острой проблему распределения исполнителей по задачам проекта.

Кроме того, в IT-отрасли производительность труда разработчика плохо поддается стандартизации и сильно варьируется от его индивидуальных особенностей. Например, время, необходимое для поиска сложной технической проблемы, часто невозможно точно оценить заранее. В связи с этим, оценки трудоемкости в этой сфере всегда должны содержать фактор неопределенности [94].

Вышеперечисленные факторы приводят к тому, что IT-проекты принципиально отличаются от других видов проектов своей неопределенностью, а также сильной зависимостью от человеческих ресурсов. Именно эти отличия формируют ключевые проблемы управления процессами проектной деятельности в IT-компании. К ним можно отнести следующие:

- проблема назначения задач исполнителям;
- проблема оценки длительности отдельных задач и проекта в целом;
- планирования времени начала выполнения каждой из задач.

Рассмотрим эти проблемы более подробно.

1.1.1 Проблема назначения задач исполнителям

Как было отмечено ранее, одной из отличительных черт IT-проектов является критическая зависимость от человеческого фактора. С учетом того, что разные исполнители имеют разный уровень квалификации, опыта, разное время решения одной и той же задачи и другие разные показатели, изменение назначений исполнителей и задач может существенно изменить показатели - проекта. Таким образом, при управлении деятельностью IT-компании обязательной является задача сопоставления задач и исполнителей. Такие задачи относятся к задачам о назначениях [58].

В настоящее время известен достаточно быстрый метод решения классической задачи (в случае отсутствия взаимной зависимости между задачами, а также наличия единственного матричного критерия оптимизации). Это хорошо известный Венгерский метод [79]. Однако, специфика предметной области требует исследования его возможностей для случая, когда имеются несколько критериев, часть их которых не сводится к матричному виду (например, балансировка нагрузки).

1.1.2 Сложность оценки сроков выполнения задач

Один из самых трудных аспектов управления проектом - это оценка длительности задач. Т.к. требования заказчика могут меняться по ходу работы, могут возникать технические проблемы, особенно в инновационных проектах, где нет аналогичного опыта. Так же, сложность задачи может быть некорректно оценена, особенно на этапе планирования. Кроме того, как было отмечено выше, длительность задачи для разных исполнителей может быть разной.

Для минимизации влияния неопределённостей и повышения точности прогнозирования сроков выполнения задач применяются различные методы и подходы, такие как, например, PERT или Монте-Карло [39]. PERT (Program

Evaluation and Review Technique) используется для анализа времени выполнения задач и их последовательности. PERT позволяет учитывать неопределённость и строить прогнозы на основе трех сценариев: оптимистичного, реалистичного и пессимистичного [22, 48]. Метод Монте-Карло представляет собой математический подход, основанный на моделировании событий с использованием случайных величин. Он помогает смоделировать множество сценариев и оценить вероятность срыва сроков [25, 62].

Сложность оценки сроков выполнения задач в IT-проектах обусловлена множеством факторов неопределённости, таких как уникальность исполнителя и его опыта, изменение требований, технические трудности и отсутствие опыта реализации подобных задач и т.д [58]. В связи с этим, вышеперечисленные методы могут быть использованы лишь в качестве основы. Требуется разработка математических моделей, которые бы учли все нюансы, связанные со спецификой данной отрасли.

1.1.3 Проблема планирования времени начала выполнения каждой из задач

В целом данная проблема относится к области сетевого планирования и управления. В частности, существует метод критического пути (Critical Path Method, CPM) [56], который позволяет получить решение классической задачи без учета ресурсов. Он базируется на временных резервах задач и позволяет определить те из них, временной сдвиг которых будет критичен для сроков выполнения проекта (так называемые, задачи, лежащие на критическом пути и имеющие нулевой временной резерв). Именно эти задачи являются приоритетными в случае, если требуется минимизировать срок выполнения проекта.

Однако, IT-проекты имеют целый ряд нюансов, требующих исследования возможностей применения данного подхода и, возможно, внесения в него существенных модификаций или разработке других

аналогов. Они связаны, в первую очередь, с уникальностью исполнителей, что влечет разный временной резерв для одной и той же задачи при разных назначениях. Кроме того, необходимо учесть динамичность IT-среды и возможность внесения изменения на разных стадиях. Это означает возможность оперативного изменения структуры план-графика, например, внепланового появления новых задач и, как следствие, возможности реагирования на такие изменения.

Таким образом, можно сделать вывод о специфичности исследуемой области, что позволяет выявить множество проблем, которые необходимо решить для эффективного управления IT-компанией как организационной системой. В частности, необходимо предложить эффективный подход для оценивания стохастических параметров, назначения исполнителям наиболее подходящих задач и планирования IT-проектов [72]. Рассмотрим и проанализируем существующие подходы, которые можно использовать для решения соответствующих задач.

1.2 Анализ существующих подходов назначения задачам специалистов

При решении задач управления персоналом часто возникает необходимость назначить исполнителей для решения однотипных задач. Проблема заключается в оптимальном распределении исполнителей по работам таким образом, чтобы максимизировать суммарный критерий эффективности или минимизировать суммарный критерий неэффективности выполнения всех работ.

Рассмотрим основные методы решения данной задачи и проанализируем их применимость для решения задачи распределения исполнителей по задачам в IT-компании.

Ручное распределение задач - это наиболее простой и традиционный метод, который полагается на опыт, интуицию и экспертные знания менеджера, который вручную распределяет задачи между исполнителями

основываясь на их квалификации, текущей загрузке, опыте и предпочтениях. К достоинствам данного подхода можно отнести скорость решения (связано с отсутствием каких-либо сложных вычислительных операций), а также, возможно, учет каких-либо неформальных факторов, которые сложно поддаются алгоритмизации (например, мотивация исполнителя, его умение работать в команде и т.д.) [58].

Недостатками такого распределения, являются высокая субъективность, неэффективность в крупных проектах и сложность учета динамических изменений, таких как задержки, изменения требований или внезапные изменения в доступности ресурсов. Кроме того, такое распределение не гарантирует даже некоторой локальной оптимизации (тем более, если будет решаться сложная оптимизационная задача с несколькими критериями). Таким образом, для исследуемой задачи в предположении о большом количестве крупных проектов данный способ будет неэффективен.

Другой вариант заключается в применении заранее подготовленных правил и шаблонов. Этот подход основан на заранее определенных правилах и стандартах распределения задач. Правила могут быть задокументированы в виде шаблонов или алгоритмов, которые применяются ко всем новым задачам. Тут есть преимущества (ускорение процесса распределения, уменьшение роли человеческого фактора) и ограничения (недостаточная гибкость, отсутствие учета динамических изменений).

Наиболее распространенным вариантом является использование классических методов оптимизации для решения задачи о назначениях. Эти методы основаны на математических моделях и алгоритмах.

Рассмотрим подробнее математическое описание задачи о назначениях. Имеется n исполнителей и n работ. Каждый исполнитель может выполнять только одну работу, а каждая работа может быть выполнена только одним исполнителем. Цель заключается в минимизации (или максимизации) суммарного критерия эффективности выполнения всех работ.

Если количество исполнителей совпадает с количеством работ (n=m), задача называется сбалансированной. В этом случае выполняются два условия:

1. Каждый исполнитель выполняет только одну работу.
2. Каждая работа выполняется только одним исполнителем.

Если количество исполнителей и работ не совпадает (n<>m), задача считается несбалансированной. Однако любую несбалансированную задачу можно привести к сбалансированной, добавляя фиктивные строки или столбцы с нулевыми или бесконечными значениями. Например, если n<m, добавляются m-n строк с нулевыми значениями, а если n>m, добавляются n-m столбцов с бесконечными (или очень большими) значениями.

Задача о назначениях может быть сформулирована как задача минимизации или максимизации. Эти две постановки эквивалентны друг другу (переход осуществляется умножением всех элементов критериальной матрицы на -1). Наиболее распространенным методом решения классической задачи является Венгерский метод. Проанализируем его с точки зрения возможности применения к рассматриваемой задаче.

Венгерский метод

Венгерский метод основывается на преобразовании исходной матрицы стоимости C в эквивалентную матрицу $C_Э$ содержащую систему независимых нулей, которые определяют оптимальное решение [58, 79].

Алгоритм направлен на поиск преобразований, которые приводят к появлению системы независимых нулей, что позволяет получить оптимальное решение задачи о назначениях.

Он состоит из последовательности шагов, каждый из которых направлен на преобразование матрицы стоимости до тех пор, пока не будет найдено оптимальное решение. Рассмотрим эти шаги подробнее. На первом и втором шагах осуществляется редукция исходной матрицы, т.е. формирование из нее матрицы, каждая строка и каждый столбец которой содержат нулевые значения.

Шаг 1. Вычитание минимального элемента из каждой строки.

Для каждой строки i матрицы стоимости C ищется минимальный элемент.

$$min_i = min_{1 \le j \le n} C[i, j] \quad (1.1)$$

Далее этот элемент вычитается из всех элементов соответствующей строки.

$$C[i, j] = C[i, j] - min_i, j = 1, \dots, n. \quad (1.2)$$

Шаг 2. Вычитание минимального элемента из каждого столбца.

Для каждого столбца j матрицы, который ещё не содержит нулевых элементов, также ищется минимальный элемент:

$$min_j = min_{1 \le i \le n} C[i, j]. \quad (1.3)$$

Он вычитается из всех элементов соответствующего столбца.

$$C[i, j] = C[i, j] - min_j, i = 1, \dots, n. \quad (1.4)$$

Шаг 3. Покрытие всех нулей минимальным количеством линий.

Проводится минимальное количество горизонтальных и вертикальных линий через строки и столбцы так, чтобы все нулевые элементы оказались покрытыми. Если количество проведённых линий равно размеру матрицы (n), то оптимальное решение найдено: на пересечении данных линий будет находиться соответствие исполнителя и работы. В противном случае осуществляется переход к следующему шагу.

С точки зрения реализации каждая вычеркнутая горизонтальная линия – это выбранная строка, а вертикальная – выбранный столбец. Пусть все выбранные строки и столбцы составляют множества Rows и Cols соответственно.

Шаг 4. Преобразование матрицы для получения новых нулей.

Находим минимальный элемент среди всех «невычеркнутых» элементов матрицы. Это означает, что индекс строки элемента не принадлежит множеству Rows, а индекс столбца – множеству Cols:

$$\min = min_{i \notin Rows;\, j \notin Cols} C[i,j]. \quad (1.5)$$

Вычитаем этот элемент из всех «невычеркнутых» элементов.

$$C[i,j] = C[i,j] - min, i = 1, \dots, n, j = 1, \dots, n. \quad (1.6)$$

Прибавляем его ко всем элементам, лежащим на пересечении проведённых линий.

$$C[i,j] = C[i,j] + min, i \in Rows, j \in Cols. \quad (1.7)$$

Элементы, через которые проходит только одна линия, остаются без изменений.

Затем проверяется, можно ли теперь построить систему независимых нулей, таким образом, что никакие два нуля не принадлежат одной и той же строке или одному и тому же столбцу.

$$\exists_p : C\big[i, p[i]\big] = 0, \forall_i \quad (1.8)$$

Если назначение невозможно, то шаги 3 и 4 повторяют до тех пор, пока оптимальное решение не будет найдено.

Венгерский алгоритм представляет собой мощный инструмент для решения задач о назначениях, позволяющий находить оптимальное решение за полиномиальное время $O(n^3)$. Его основные преимущества заключаются в простоте реализации и высокой эффективности, особенно для задач с большим количеством исполнителей и работ. Математические формулы, описывающие этапы алгоритма, делают его универсальным и применимым для широкого класса задач, связанных с минимизацией или максимизацией целевых функций.

Проанализируем возможности его применения относительно исследуемой задачи. В первую очередь, необходимо отметить, что Венгерский метод максимизирует или минимизирует единственный критерий. Следовательно, возникает задача агрегирования критериев, что не всегда возможно из-за слишком разной природы данных критериев. Во-

вторых, Венгерский метод предполагает фиксированный набор задач и исполнителей и является эффективным для статичных задач. Однако, для решения задачи о назначении задач исполнителям в IT-проектах необходимо учитывать динамику проекта (появление новых задач, изменение их приоритетов, уход или временное отсутствие исполнителей и т.д.). Кроме того, желательно учесть уникальность исполнителей.

Таким образом, хотя венгерский алгоритм является надёжным инструментом для решения задач о назначениях в условиях статичности и определённости, его использование в IT-проектах требует дополнительной адаптации или комбинирования с другими методами, чтобы учитывать специфику динамичной и неопределённой среды IT-разработки.

Эволюционные методы

В случае отклонения исследуемой задачи от классического варианта с одинаковым числом строк и столбцов и единственным критерием, для решения задачи о назначении могут применяться эволюционные методы. Они представляют собой мощный инструмент, позволяющий находить субоптимальные решения для задач любой сложности. Проанализируем возможность применения данных методов для исследуемой задачи, их достоинства и недостатки.

Одним из наиболее распространенных эволюционных методов, применяющихся в оптимизационных задачах большой сложности, является генетический алгоритм [10,11]. Основная идея генетических алгоритмов основана на поиске наилучших решений путем создания новой популяции особей и сравнении каждой из них с эталонным значением. Каждая такая особь-потомок получается путем скрещивания двух родителей предков и формируется путем добавления генов как от первого, так и от второго предка. Для того, чтобы избежать «застревания» в локальных оптимумах, к популяции применяется процедура мутации. С некоторой вероятностью выбирается особь (одна или несколько), в гены которой вносятся

определенные изменения. Таким образом, в целом генетический алгоритм состоит из следующих шагов:

1. Формирование начальной популяции, определение наилучшего (эталонного) элемента.

2. В цикле по числу итераций:

2.1. В цикле по числу потомков

2.1.1. Выбираются два родителя

2.1.2. Осуществляется процедура скрещивания

2.2. Вероятностным образом осуществляется мутация

2.3. Производится сравнительный анализ новой популяции, в результате которого:

- проверяется, получена ли особь, лучше эталонной с точки зрения фитнес-функции (если это так, то эталонное значение обновляется);

- отсеивается некоторое количество наихудших с точки зрения фитнес-функции особей.

Применительно к исследуемой задаче хромосому целесообразно кодировать в виде массива, каждый элемент которого содержит следующие значения [75]:

- задача;

- исполнитель;

-время начала;

- длительность.

Выбор особей для скрещивания можно выполнить одним из стандартных подходов (случайным, турнирным, методом рулетки и т.д.) [10, 11]. В качестве операции скрещивания можно предложить следующий вариант [75]:

- случайным образом выбирается задача;

- последовательность назначений до данной задачи берется от родителя 1; после – от родителя 2 (при этом, как правило, переопределяются

длительнсти во второй части из-за необходимости учета нового первоначального расписания).

Мутация может заключаться в обмене задачами для двух случайно выбранных исполнителей и дальнейшем переопределении характеристик проекта.

Применительно к данной задачи следует выделить следующие преимущества генетического алгоритма:

- возможность эффективно учесть несколько критериев, что является важным в исследуемой области;

- генетический алгоритм в меньшей степени склонен к «застреванию» в локальных оптимумах;

- генетические алгоритмы достаточно гибкие, их можно подстроить под данную задачу с любыми вариациями через специфические операторы скрещивания и мутации.

Основным минусом генетического алгоритма является его вычислительная сложность. Оценка фитнес-функции для одной особи требует симуляции расписания проекта (расчета критического пути с учетом назначений и зависимостей), что является дорогостоящей операцией. Выполнение ее для всей популяции в сотни поколений может быть неприемлемо медленно для больших проектов. Другим существенным минусом генетических алгоритмов является то, что в результате скрещивания и мутации могут появляться недопустимые решения. Следовательно, необходима дополнительная проверка и существенное увеличение шагов цикла, что, в конечном итоге еще сильнее увеличит сложность алгоритма. Третий недостаток заключается в необходимости проведения множества экспериментов для выбора наилучшего способа скрещивания и мутации.

Таким образом, вычислительная затратность и сложность настройки делают генетический алгоритм менее привлекательным для частого перепланирования в динамичной среде IT-проекта.

Другим вариантом эволюционных методов являются муравьиные алгоритмы [95]. Основная идея этого метода заключается в поиске наилучшего решения путем имитации работы колонии муравьев. Каждый муравей случайным образом выбирает некоторое решение. Однако, на вероятность такого выбора оказывает влияние уровень ферромона, оставленный на данном решении муравьями на предыдущих этапах. Кроме того, эта вероятность будет зависеть от качества данного решения (значения фитнес-функции в данной точке). После каждого этапа осуществляется пересчет уровня ферромона: на выбранном решении он увеличивается, на остальных уменьшается. Параллельно на каждом этапе осуществляется сравнительный анализ текущего решения с эталоном с целью поиска оптимума.

Применительно к исследуемой задаче она будет представлена в виде графа, соответствующего сетевому графику. Муравей, продвигаясь по нему, будет последовательно назначать задачи исполнителям.

В качестве достоинств муравьиного алгоритма следует выделить естественное представление для данной задачи, учет положительной обратной связи, а также возможное сочетание с другими подходами (например, с эвристиками) в вероятностном выборе соответствия задачи исполнителю.

К недостаткам муравьиных алгоритмов необходимо отнести также большие вычислительные ресурсы, поскольку построение расписания для каждого муравья и последующий пересчет для каждого решения является достаточно трудоемким процессом. Вторым недостатком является параметрическая сложность: муравьиный алгоритм зависит от большого числа параметров (параметры α и β для выбора вероятности, параметр ρ, регулирующий скорость испарения и ряд других). Кроме того, применительно к данной задаче недостатком будет являться ее многокритериальность: требуется сложная агрегация критериев, что ведет к тем же проблемам с весами, что и в классических методах.

В связи с этим, можно сделать вывод о том, что метод муравьиных колоний хорошо подходит для задач комбинаторной оптимизации с явно выраженной графовой структурой. Однако его применение для многокритериальной задачи с «тяжелой» функцией оценки является нетривиальным и может не дать явных преимуществ перед другими методами.

Крайне быстрым (по сравнению с двумя рассматриваемыми методами) является алгоритм имитации отжига. Его суть заключается в последовательном сравнении текущего решения с одним из его «соседей». Если новое решение лучше старого, то оно берется в качестве эталона. В противном случае принятие решения носит вероятностный характер, причем данная вероятность зависит от разности между фитнес-функциями нового и старого решения, а также текущей температуры. Метод работает в цикле от так называемой максимальной до минимальной температуры. Чем ниже температура, тем меньше вероятность выбора худшего решения.

К плясам данного решения следует отнести, в первую очередь, скорость. В связи с этим, у метода крайне низкие требования к вычислительным ресурсам. Вторым достоинством является простота реализации. Для работы метода требуется определить только представление решения, функцию соседства и функцию оценки. Еще одно преимущество заключается в способности выходить из локальных оптимумов за счет вероятностного механизма принятия более худших решений.

К недостаткам метода следует отнести его существенную зависимость от начального решения. Если начальная точка будет далека от оптимальной, вероятность получить оптимальные результат будет невелика. Однако, если использовать данный метод на финальной стадии поиска, когда некоторый претендент на решение найден, эффективность данного метода существенно возрастает.

Таким образом, имитация отжига является хорошим вариантом решения исследуемой задачи на финальной стадии для возможного улучшения найденного минимума.

Методы машинного обучения для задач распределения работ

В современных условиях эффективное распределение задач среди исполнителей становится всё более сложной задачей, что обусловлено ростом объёмов данных и увеличением сложности проектов. В связи с этим возрос интерес к использованию методов машинного обучения (Machine Learning, ML), позволяющих проводить глубокий анализ исторических данных, выявлять нелинейные закономерности и прогнозировать оптимальные стратегии распределения задач.

Методы машинного обучения позволяют учитывать множество переменных факторов, таких как текущая загруженность исполнителей, уровень их квалификации, ожидаемые сроки выполнения и требования к качеству [1]. Обучение моделей происходит на основе накопленных данных, отражающих как успешный опыт реализации проектов, так и проблемы, возникавшие в ходе предыдущих распределений. Методы машинного обучения можно разделить на методы обучения с учителем, без учителя и с подкреплением. С точки зрения рассматриваемой задачи методы обучения с учителем предполагают распределение задач на основе некоторого набора данных, которые содержат вход и выход. В качестве входа будут выступать все особенности задачи, описанные ранее, а на выходе будет итоговое назначение исполнителей для каждой из задач. Примером такого элемента входных данных будет проект, состоящий из заданного числа задач с заданными зависимостями и длительностями, исполнители с их особенностями. На выходе будет соответствие задачи и исполнителя. Классические методы машинного обучения с учителем включают в себя методы прогнозирования (разнообразные регрессионные модели) и методы решения задач классификации, что не подходит для решения исследуемой

задачи. Основные алгоритмы регрессии включают в себя линейную и логистическую регрессию, деревья решений и нейронные сети.

Проанализируем возможности нейронных сетей для данной задачи. С учетом того, что на входе сети должен быть граф зависимостей и многомерные (возможно, матричные) характеристиками исполнителей, стандартные полносвязные сети не подойдут, поскольку они не учитывают структуру IT-проекта [56]. Необходима архитектура, которая способна работать с графами и комбинировать информацию из двух разных источников: узлов графа (задачи) и исполнителей. В связи с этим, целесообразно использовать графовые нейронные сети (GNN) [1]. Входные слои должны содержать энкодер для задач, позволяющий преобразовать граф и его контекст в векторное представление. Кроме того, потребуется также преобразование характеристик каждого исполнителя в вектор. Далее должны быть сформированы внутренние слои, где задачи так или иначе выбирают себе исполнителей (возможно вероятностное распределение) и наоборот. Таким образом, внутри сети должно определиться наилучшее взаимодействие между задачами IT-проекта и исполнителями. При этом финальное распределение должно быть валидным с точки зрения ограничений задачи (для одной задачи используется только 1 исполнитель и т.д.).

Для обучения такой сети необходим набор данных об эффективных назначениях исполнителей задачам.

Проанализируем недостаток такого подхода. Во-первых, для обучения нейронной сети необходима выборка очень большого объема. Следовательно, надо иметь большое количество проектов с достаточно эффективным распределением исполнителей по задачам. Получить такую выборку достаточно затруднительно с учетом того, что данная область постоянно меняется. Кроме того, проекты достаточно большие. В связи с этим, IT-компания не сможет накопить необходимое количество данных (на это должны уйти годы, а другие компании, очевидно, не будут делиться

данной информацией в связи с конфиденциальностью данных). Следовательно, очень сложно подобрать тысячи проектов с успешным распределением, исходные данные которых будут содержать всю необходимую информацию (поскольку со временем она меняется).

Выходом из данной ситуации является использование обучения с подкреплением. Основная идея подобных методов заключается в возможности принятия решений, находясь в некоторой среде, имитирующей реальную ситуацию, и получая обратную связь от каждого принимаемого решения в виде поощрения или наказания. Применительно к решаемой задаче в качестве среды будет выступать граф задач и возможные исполнители. Действие будет заключаться в назначении конкретного исполнителя на конкретную задачу. Награда это рассматриваемая целевая функция (например, общее время выполнения всего проекта или иная метрика, которую необходимо минимизировать).

К несомненным достоинствам данного подхода будет относиться то, что качественно натренированная модель может очень быстро (за одно прямое распространение) выдать решение для новой, похожей конфигурации задач и исполнителей. Кроме того, модель может обучаться на сложных нелинейных функциях награды (что будет актуально, например, в случае многокритериальной оптимизации) [49].

Однако, обучение с подкреплением требует огромного количества эпизодов для обучения (много симуляций или исторических данных). Кроме того, IT-проект в целом и исследуемая задача в частности характеризуется большим числом гиперпараметров, что делает работу методов обучения с подкреплением весьма нестабильными [30, 56]. В связи с этим, возможно их использование для других частных задач управление деятельностью IT-компании, а для решения задачи о назначениях с такой структурой их применение нецелесообразно.

Гибридные подходы.

В целях повышения эффективности и адаптивности систем распределения задач в условиях многозадачности и высокой динамики проектной среды, в последние годы возрастающее внимание уделяется гибридным подходам. Эти подходы представляют собой интеграцию различных методов, от экспертных правил до алгоритмов оптимизации и машинного обучения, с целью достижения баланса между скоростью, точностью и устойчивостью решений.

Гибридные стратегии особенно востребованы в сфере информационных технологий, где требуется учитывать множество переменных: быстрое изменение требований, ограниченность ресурсов, приоритеты заказчиков, компетенции исполнителей и временные ограничения. Сочетание подходов позволяет не только повышать общую эффективность, но и обеспечивать большую прозрачность и управляемость процессов назначения задач.

Типовой пример реализации гибридного подхода может включать следующие этапы.

Эвристическая и экспертная фаза: на этом этапе используются заранее заданные правила, шаблоны и предпочтения, позволяющие быстро сформировать предварительное распределение задач. Это позволяет оперативно реагировать на изменения и минимизировать влияние человеческого субъективного фактора.

Оптимизационная фаза: применяется математическое моделирование, включая венгерский алгоритм, методы линейного и целочисленного программирования. Эти инструменты обеспечивают точное распределение ресурсов с учётом множества ограничений и критериев оптимальности.

Фаза машинного обучения и адаптации: на завершающем этапе задействуются обученные ML-модели, анализирующие исторические данные и текущие условия. Они способны предсказывать последствия назначений и

адаптировать стратегию распределения в реальном времени, повышая устойчивость системы к неопределённости.

Преимущества гибридных подходов включают в себя повышенную адаптивность и масштабируемость, снижение зависимости от одного метода и, как следствие, снижение риска сбоев, возможность непрерывного совершенствования за счёт внедрения обратной связи и машинного обучения, эффективное распределение задач как в условиях дефицита ресурсов, так и при высокой степени неопределённости.

Ключевым вызовом при разработке и внедрении гибридных систем остаётся необходимость согласования различных компонент подхода, включая формальные алгоритмы, модели данных и интерфейсы взаимодействия между автоматизированными и человеческими агентами.

Таким образом, гибридные модели представляют собой перспективное направление в эволюции систем управления задачами, объединяя вычислительную строгость оптимизационных методов, адаптивность машинного обучения и прагматизм экспертных систем. Их развитие требует мультидисциплинарного подхода, сочетающего знания в области проектного управления, прикладной математики и интеллектуального анализа данных.

Сравнение существующих подходов к задаче назначения работам специалистов можно посмотреть в Таблице 1.1.

Таблица 1.1 - Сравнение существующих подходов к задаче назначения работам специалистов

Критерий	Ручное распределение	Оптимизационные модели	Машинное обучение
Точность	Низкая, зависит от опыта менеджера	Высокая, учитывают множество факторов	Очень высокая, анализирует исторические данные
Гибкость	Высокая, легко адаптируется к изменениям	Средняя, требуют обновления при изменениях	Очень высокая, адаптируется к новым данным
Вычислительная сложность	Минимальная, не требует вычислений	Высокая, особенно для сложных задач	Очень высокая, требует вычислительных ресурсов
Применимость	Малые проекты с ограниченным числом задач	Проекты со средним числом задач и ограничений	Крупные проекты с большими объемами данных

Таким образом, подводя итог анализа существующих методов назначения исполнителей для каждой из задач IT-проекта можно сделать следующие выводы. Существующие методы назначения исполнителей, применяемые на практике или известные из теории (включая классические алгоритмы типа Венгерского), неприменимы в чистом виде для решения комплексной многокритериальной задачи распределения исполнителей по задачам в IT-проектах. Их фундаментальные ограничения (наличие только одного критерия, отсутствие учета влияния назначений на сроки проекта через критический путь, игнорирование необходимости балансировки загрузки и множества критериев качества, статичность) делают их решения неоптимальными или даже нежизнеспособными в условиях уникальности IT-проектов (высокая неопределенность, зависимость от человеческого фактора, динамика).

В связи с этим, необходима модификация существующих или разработка новых подходов к решению данной задачи, которая бы отвечала следующим требованиям:

- учитывала бы совместное решение данной задачи с задачей планирования (в частности, предусматривала бы параллельное решение задачи планирования с точки зрения критерия скорейшего завершения проекта);

- позволяет реализовать многокритериальный подход для одновременной оптимизации критериев назначения исполнителей задачам, балансировки нагрузки исполнителей и оптимизации времени завершения проекта;

- обладала бы достаточной гибкостью для практического применения в динамической среде IT-компаний.

1.3 Анализ существующих подходов к проблеме оценки сроков выполнения проектов

Оценка сроков выполнения проектов является одной из ключевых задач управления IT-проектами, так как ошибки в прогнозах могут приводить к серьёзным последствиям: срыву сроков, перерасходу бюджета и снижению удовлетворённости клиентов [11, 54, 86]. Современные подходы к оценке сроков выполнения проектов можно разделить на следующие категории: экспертные оценки, классические методы решения задач сетевого планирования и управления (в первую очередь, PERT), а также симуляционные методы. Проанализируем их более подробно.

Экспертные оценки.

Экспертные оценки - это один из наиболее широко используемых методов, основанный на субъективных прогнозах опытных специалистов. Суть метода заключается в привлечении экспертов для оценки временных, трудовых или стоимостных параметров выполнения задачи. Эксперты,

опираясь на свой опыт и знания из прошлых проектов, предоставляют прогнозы, которые затем анализируются и обобщаются для принятия решений.

Экспертные оценки имеют несколько преимуществ: они просты и быстро осуществимы, позволяют учесть практический опыт и интуицию специалистов, дают возможность услышать мнение независимых экспертов и провести коллективное обсуждение спорных вопросов.

Но этот метод так же имеет и ограничения - высокая субъективность, невозможность учёта всех факторов неопределённости, ограниченная масштабируемость, риск группового мышления.

Метод экспертных оценок особенно полезен на ранних этапах проекта, когда недостаточно данных для применения более формализованных методов, таких как оптимизационные модели или машинное обучение.

После получения экспертных оценок их можно использовать как входные данные для других методов, таких как метод PERT (Program Evaluation and Review Technique) или критический путь , чтобы повысить точность прогнозов.

Метод PERT

Метод PERT (Program Evaluation and Review Technique) - это инструмент управления проектами, который позволяет оценивать сроки выполнения задач с учетом неопределенности. В отличие от метода критического пути, который предполагает фиксированное время выполнения задач, PERT использует вероятностный подход, основанный на трех временных оценках: оптимистической, пессимистической и наиболее вероятной. Для каждой задачи i оцениваются три временных параметра: a_i - оптимистическое время выполнения (минимально возможное время), b_i - пессимистическое время выполнения (максимально возможное время), m_i - наиболее вероятное время выполнения (мода).

Среднее время выполнения задачи $dlit_i^*$ рассчитывается по формуле (1.13) [14, 15]:

$$dlit_i^* = \frac{a_i+4m_i+b_i}{6} \quad (1.9)$$

Данная формула является приближенным значением математического ожидания случайной величины, распределенной по закону бета в интервале (a_i ,b_i) с модой m_i. Было экспериментально подтверждено, что формула (1.9) достаточно точно оценивает математическое ожидание случайной величины, распределенной по закону бета [42, 56]. Точное значение математического ожидания соответствующей случайной величины имеет вид:

$$M\xi = a + \frac{p}{p+q}(b-a), \quad (1.10)$$

где [a,b] -интервал распределения случайной величины, распределенной по закону бета; p и q – параметры этой величины. С учетом того, что параметры p и q распределения изначально неизвестны, данная формула существенно упрощает оценку длительности работ без снижения качества.

Также метод PERT позволяет оценить длительность всего проекта. Для этого необходимо предварительно (например, с помощью метода CPM) построить предварительный план проекта, рассчитать раннее и позднее время начала каждой из работ и ее резерв. На основании резервов определяется критический путь как путь в графе, построенным на основании сетевого графика, каждая работа которого имеет нулевой резерв. На основании оцененных значений длительности каждой из работ PERT предлагает оценивать длительность всего проекта как сумму оценок длительностей работ, стоящих на критическом пути:

$$T^*_{\text{крит}} = \sum_{i \in Krit} dlit_i^*. \quad (1.11)$$

Кроме того, PERT аналогичным образом оценивает дисперсию проекта:

$$D^* = \sum_{i \in Krit} D_i^*. \quad (1.12)$$

Здесь D_i^* - дисперсия работы i, стоящей на критическом пути. Для бета распределения она определяется по формуле:

$$D = {(b-a)^2 \cdot p \cdot q}\Big/{((p+q)^2 \cdot (p+q+1))}. \quad (1.13)$$

Здесь [a,b] – границы интервала распределения случайной величины бета, p и q – неизвестные параметры этой случайной величины. Из-за определенной сложности оценки неизвестных параметров, для оценки дисперсии случайной величины, определяющей длительность выполнения отдельной работы в методе PERT используется правило трех сигм. Классическая постановка данного правила говорит о том, что практически все значения нормально распределенной случайной величины лежат в области, ограниченной слева и справа отрезком, величиной 3σ, где σ - среднеквадратическое отклонение. В методе PERT как эмпирическое допущение также принято данное правило. Оно существенно упрощает расчеты в силу того, что интервал [a,b] изначально предполагается известным. Исходя из этого, среднее квадратическое отклонение будет определяться по формуле:

$$\sigma = {(b-a)}/{6}. \quad (1.14)$$

Из формулы (1.14) легко найти дисперсию отдельной работы:

$$D_i^* = {(b_i - a_i)^2}/{36}. \quad (1.15)$$

Подставив (1.15) в формулу (1.12), можно найти общую дисперсию проекта.

Далее в предположении о нормальности случайной величины, описывающей весь проект, и правиле трех сигм, можно получить интервальные оценки длительности проекта:

$$\begin{cases} min = T^*_{\text{крит}} - 3\sigma; \\ max = T^*_{\text{крит}} + 3\sigma. \end{cases} \quad (1.16)$$

Здесь σ - корень из дисперсии проекта, которая определяется формулой (1.12).

Таким образом, метод PERT дает математический аппарат, который позволяет оценить стохастические характеристики проектов со случайной длительностью выполнения отдельных работ [56].

Проанализируем возможность применения метода PERT к исследуемой задаче. Отметим еще раз, что особенностью данной отрасли является возможность коррекции задач в следствие ошибок, совершаемых по разным причинам (недостаточная квалификация, новые технологии, не до конца понятное техническое задание и т.д.), а также в следствие недопонимания заказчика или изменения его требований, например, в ходе тестирования или в силу каких-то других причин. Очевидно, что исследуемая случайная величина уже должна вести себя иначе, чем классический закон бета. В связи с этим, необходимо изучить специфику такого поведения с целью коррекции аппарата PERT для решения задачи оценивания параметров.

Симуляционные методы

Данная категория методов позволяет оценить искомые случайные величины путем их многократного розыгрыша. Среди них наиболее популярным является метод Монте-Карло, который позволяет проводить тысячи симуляций выполнения проекта с учётом различных сценариев.

Основной принцип заключается в использовании большого количества случайных испытаний для приближения вероятностного распределения искомой величины [25].

Для этого нужно пройти следующие шаги:

- определить задачу (формулируется проблема, которую необходимо решить с помощью моделирования);

- сгенерировать случайные числа (используются генераторы чисел для создания выборок) [30];

- смоделировать испытания (проводится множество экспериментов или симуляций с использованием сгенерированных данных) [56, 62];

- проанализировать результаты (собираются результаты всех испытаний, на основе которых строится вероятностное распределение; выводятся статистические показатели - среднее значение, дисперсия и другие параметры интереса).

Плюсы метода Монте-Карло:

- простота реализации (метод не требует сложных математических преобразований или глубоких знаний в области теории вероятностей);

- универсальность (метод Монте-Карло можно применять к широкому спектру задач: от моделирования физических процессов до оценки финансовых рисков);

- работа с неопределенностью (Монте-Карло справляется с задачами, где присутствует высокая степень неопределенности или вариативности данных; он позволяет моделировать различные сценарии развития событий, что особенно полезно при прогнозировании);

- параллельные вычисления (этот метод идеально подходит для параллельных вычислений, что значительно ускоряет процесс обработки данных на современных многопроцессорных системах).

Минусы метода Монте-Карло:

- высокие вычислительные затраты (высокая требовательность к вычислительным ресурсам, особенно при необходимости проведения большого количества симуляций);

- необходимость наличия исторических данных для построения моделей.

Анализируя специфику данного метода для исследуемой задачи, можно констатировать, что в интеграции с методом PERT он может быть полезен для оценки стохастических характеристик IT-проекта с учетом возможной коррекции задач, появления новых или модификации существующих задач.

1.4 Анализ подходов к планированию времени начала выполнения задач

Среди методов, позволяющих оценить время начала каждой из задач, стоит в первую очередь выделить метод критического пути (Critical Path Method, CPM). Это инструмент управления проектами, который позволяет определить последовательность задач, оказывающих наибольшее влияние на сроки завершения проекта. CMP используется для планирования, составления расписания и контроля проектов, особенно в условиях, где важно соблюдение временных рамок.

Основные этапы построения сетевой диаграммы CPM

1. Разбиение проекта на виды работ. Проект разбивается на отдельные задачи или операции, которые необходимо выполнить. Каждая задача получает уникальное обозначение (например, буквенное или числовое). Перечень задач служит основой для дальнейшего анализа.

2. Определение последовательности действий. Устанавливаются взаимосвязи между задачами: какие из них должны быть выполнены до начала других. Это создает логическую цепочку выполнения работ, которая отражает технологическую последовательность проекта.

3. Построение сетевой диаграммы. Задачи представляются в виде узлов (кружков), а их зависимости - в виде стрелок или линий. Сетевая диаграмма наглядно демонстрирует структуру проекта и взаимосвязь всех операций.

4. Оценка продолжительности задач. Для каждой задачи i определяется время выполнения t_i. Это можно сделать на основе исторических данных, экспертных оценок или аналогичных проектов. Продолжительность задач указывается на сетевой диаграмме.

5. Определение критического пути. На этом этапе вычисляется максимально возможная продолжительность последовательно выполняемых работ. Критический путь - это самая длинная цепочка задач с точки зрения времени выполнения, задержка в которой приведет к задержке всего проекта. Пример сетевой диаграммы представлен на Рисунке 1.1.

6. Обновление сетевой диаграммы. По мере реализации проекта данные о времени выполнения задач могут изменяться. Сетевая диаграмма обновляется, чтобы учесть новые условия, и может появиться новый критический путь.

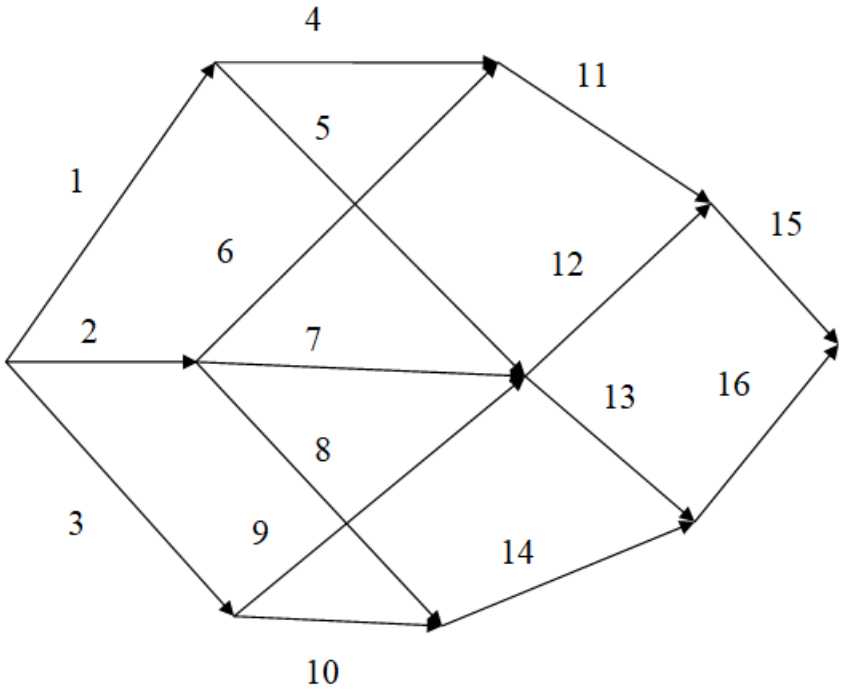


Рисунок 1.1 - пример сетевой диаграммы CPM

Для определения критического пути используются следующие параметры:

ES (Early Start) - ранний старт: самое раннее время, когда задача может начаться.

$$ES_j = \max(ES_i + t_i) \quad (1.17)$$

где ES_i - ранний старт предшествующих задач i, t_i - их продолжительность.

EF (Early Finish) - ранний финиш: самое раннее время, когда задача может завершиться.

$$EF_j = ES_j + t_j \quad (1.18)$$

где t_j - продолжительность задачи j.

LS (Late Start) - позднее начало: самое позднее время, когда задача может начаться без нарушения сроков проекта.

$$LS_j = LF_j - t_j \quad (1.19)$$

LF (Late Finish) - позднее окончание: самое позднее время, когда задача может завершиться без нарушения сроков проекта.

$$LF_j = \min(LF_k + t_k) \quad (1.20)$$

где LF_k - позднее окончание последующих задач k.

Критический путь определяется как путь, на котором:

$$ES_j = LS_j \quad (1.21)$$

$$EF_j = LF_J \quad (1.22)$$

Это означает, что у задач на критическом пути нет резерва времени (SL, Slack):

$$SL_j = LS_j - ES_j = LF_j - EF_j \quad (1.23)$$

Расчет общего времени завершения проекта

Общее время завершения проекта T определяется как сумма продолжительностей задач на критическом пути:

$$T = \sum_{j \in CP} t_j \quad (1.24)$$

где CP - множество задач, принадлежащих критическому пути.

Метод CPM имеет следующие преимущества:

– графическое представление проекта. наглядная сетевая диаграмма помогает понять структуру проекта и взаимосвязь задач;

– определение сроков выполнения задач. cpm показывает, когда каждая задача должна начинаться и заканчиваться;

– вычисление общего времени завершения проекта (метод позволяет рассчитать минимальное время, необходимое для завершения проекта);

– идентификация критических задач (помогает выделить задачи, которые напрямую влияют на сроки проекта, и сосредоточить на них внимание).

В тоже время у этого метода существуют ограничения, которые заключаются в:

– отсутствии учета колебаний времени выполнения задач (метод предполагает фиксированное время выполнения задач, что не всегда соответствует реальности, особенно в сложных проектах).

– частичная применимость для уникальных проектов (cpm лучше работает с повторяющимися задачами, где время выполнения можно точно спрогнозировать. в уникальных или инновационных проектах метод может оказаться менее эффективным).

– сложность применения в крупных проектах (для больших проектов с множеством задач построение и анализ сетевой диаграммы может стать трудоемким процессом).

CPM наиболее эффективен для проектов с четко определенными задачами и фиксированным временем выполнения.

Однако для проектов с высокой степенью неопределенности или уникальных задач (например, разработка новых технологий), к классу которых относятся IT-проекты, метод CPM следует комбинировать с другими подходами, такими как PERT (Program Evaluation and Review Technique) или гибкие методологии (Agile), чтобы учесть стохастические факторы и адаптироваться к изменениям.

1.5 Цель и задачи диссертационного исследования

Целью работы является разработка моделей и алгоритмов управления деятельностью IT-компании как организационной системы путем определения наилучших сроков для отдельных задач IT-проекта, допускающих возможность их повторного выполнения или коррекции, а также назначения им исполнителей в зависимости от их квалификации.

Задачи исследования. Для достижения поставленной цели необходимо решить следующие задачи:

1. Провести анализ существующих подходов к управлению процессами проектной деятельности в IT-компании как организационной системы в рамках поиска наилучшего подхода к управлению коллективом исполнителей, распределению задач и планирования проектов.

2. Разработать модель управления проектом в IT-компании как организационной системе, отличающуюся учетом квалификации специалистов, их занятости, типизации задач и обеспечивающей наилучшую организацию деятельности IT-компании путем планирования времени начала каждой из задач проекта и подбора для нее исполнителя.

3. Разработать математическую модель оценки сроков выполнения как отдельных задач, так и всего IT-проекта данной организационной системы, отличающуюся учетом коррекции задач, обеспечивающую повышенную точность по сравнению с аналогами.

4. Разработать алгоритмы организации работ в IT-компании, отличающиеся одновременным решением задач определения сроков и времени начала для каждого из заданий IT-проекта и подбора для них наилучших исполнителей и обеспечивающие возможность повышения качества управления деятельностью IT-компании за счет более рационального использования человеческих ресурсов и более точных оценок времени выполнения как всего проекта, так и его отдельных заданий.

5. Реализовать программное обеспечение системы управления IT-компанией как организационной системой, отличающееся интеграцией систем имитационного моделирования, планирования и управления и базы данных, и обеспечивающего оптимизацию деятельности IT-компании с точки зрения выбора оптимальных сроков решения каждой из задач IT-проекта и подбора наилучшего исполнителя для нее [72].

6. Провести экспериментальное исследование эффективности разработанных моделей и алгоритмов управления IT-компанией как организационной системой с точки зрения длительности IT-проекта.

Дизайн исследования представлен на Рисунке 1.2.

Рисунок 1.2 – Дизайн исследования

ГЛАВА 2. МОДЕЛИРОВАНИЕ ПРОЦЕССА УПРАВЛЕНИЯ IT-ПРОЕКТОМ С УЧЕТОМ КВАЛИФИКАЦИИ И ЗАНЯТОСТИ СПЕЦИАЛИСТОВ И ВОЗМОЖНОЙ КОРРЕКЦИИ ЗАДАЧ

В главе рассматриваются задачи построения математических моделей, позволяющих описать процесс управления IT-проектом с учетом разной квалификации исполнителей, а также их занятости и оценки длительности IT-проекта и его отдельных задач. В первой части предложена общая модель управления IT-проектом, отличающаяся учетом квалификации специалистов, их занятости, типизации задач, а также многокритериальная оптимизационная задача управления путем назначения задачам исполнителей и формирования план-графика. Вторая часть посвящена построению математической модели, оценки сроков выполнения отдельных задач, отличающуюся учетом коррекции задач и обеспечивающую повышенную точность по сравнению с аналогами. В третьей части описывается процесс построения имитационной модели, позволяющей оценить длительность случайной величины, описывающей длительность всего проекта, и ее вероятностно-временные характеристики, допускающую коррекцию отдельных задач. Выводы приведены в четвертой части диссертации.

2.1 Формализация задачи управления IT-проектом в условиях разной квалификации исполнителей и возможной коррекции задач

Рассмотрим процесс управления IT-проектом, ключевыми особенностями которого является наличие команды исполнителей разной квалификации. Целью является разработка математической модели, позволяющей учесть квалификацию специалистов, их занятость, а также типизации задач. Задачу можно сформулировать следующим образом. Пусть имеется K групп специалистов, осуществляющих процесс выполнения

заданий IT-проекта, каждый из которых имеет определенную специализацию и квалификацию. Численность групп составляет M_1, M_2,…, M_K исполнителей соответственно. Квалификация специалистов отражается на времени решения задач и качестве получаемых решений. На вход поступает проект, представляющий собой множество взаимно-зависимых задач (пусть, без ограничения общности, их будет N), каждая из которых относится к одному из K типов. Время решения задачи является случайной величиной, однако, каждый специалист характеризуется индивидуальными параметрами данной величины. Известно ориентировочное расписание занятости исполнителей задачами, поступившими ранее. Требуется выполнить соответствие между поступившими задачами и исполнителями с точки зрения целей, предпочтительных для лица, принимающего решение и время начала выполнения каждой задачи.

Математическая модель будет строиться при следующих допущениях:

- случайные величины ξ_{ij}, описывающие длительность выполнения задачи j специалистом i, могут быть оценены наименьшим, наибольшим и наиболее вероятным значениями;

- после окончания выполнения задачи она может потребовать по разным причинам коррекции, которая для специалиста, выполняющего ее, может быть оценена некоторой вероятностью;

- для данного специалиста также можно оценить наименьшее, наибольшее и наиболее вероятное время исправления работы (а также, возможной повторной коррекции);

- специалисты имеют некоторое расписание своей занятости, которое корректируется при назначении им работ и завершения их выполнения и учитывается при решении задач назначения и планирования.

Отметим, что для каждой группы специалистов (для каждого типа задач) можно сформировать модель, которая будет зависеть от других моделей лишь моментами окончания и начала взаимно зависимых задач. Однако, это даст возможность распараллелить алгоритм, что, в свою очередь

существенно сократит время, затрачиваемое на решение. Рассмотрим модель для группы k (k=1, …, K).

Введем следующие обозначения:

M_k – число специалистов в данной группе;

N_k- число задач в проекте у данной группы;

K – число типов задач (и число специализаций исполнителей).

Требуется найти матрицу x, которая определяется формулой:

$$x_{ij} = \begin{cases} 0, \text{исполнитель } j \text{ не выполняет задачу } i; \\ 1, \text{ исполнителю } j \text{ назначена задача } i. \end{cases} \quad (2.1)$$

Иными словами, требуется каждой задаче поставить в соответствия исполнителя. Кроме того, требуется определить:

- время начала выполнения каждой задачи i $t_{\text{нач}}(i)$;

- оценить сроки выполнения всего проекта.

Добавим классическое ограничение задачи о назначениях, заключающееся в том, чтобы каждая задача решалась лишь одним специалистом:

$$\sum_{j=1}^{M_k} x_{ij} = 1 \text{ , i=1, …, } N_k, \quad (2.2)$$

Где

$$x_{ij} \in \{0,1\}. \quad (2.3)$$

Каждый исполнитель j задается следующими характеристиками:

-расписанием задач, запланированных ранее (на предыдущих этапах или для других проектов);

- квалификацией.

Квалификация исполнителей определяется:

- длительностью решения каждой из задач, которая у специалистов разного уровня разная;

- множеством других матричных критериев (например, качеством кода, вероятностью его безошибочного написания и т.д.).

Пусть, без ограничения общности, существует L критериев $Qual_1$, …, Qual, каждый из которых описывает определенный для ЛПР. Часть из них может быть матричными. Некоторые критерии могут быть более сложными и нелинейными (например, балансировка нагрузки исполнителей), которая определяется формулой:

$$L = \sum_{j=1}^{M} \left(L_j - \bar{L}\right)^2, \tag{2.4}$$

Где $\bar{L}$ – средняя загрузка; L_j – загрузка исполнителя j. Она будет определяться по формуле:

$$L_j = \sum_{i=1}^{N} x_{ij} \cdot dur_{ij}. \tag{2.5}$$

Здесь назначенные задачи берутся за какой-либо период.

Необходимо найти наилучшего с точки зрения всех этих критериев специалиста для каждой из задач.

Будем допускать возможную недееспособность исполнителя в определенные моменты. Для этого, а также для учета расписания введем в рассмотрение некоторую функцию:

$$Sch_j(t) = \begin{cases} 0, \text{если спец.} j \text{ не занят в момент } t; \\ 1, \text{спец} - \text{ту } j \text{ запланирована задача в момент } t; \\ 2, \text{спец} - \text{ту } j \text{ нельзя планир. задачи в момент } t. \end{cases} \tag{2.6}$$

Далее опишем задачи в проекте. Каждая задача имеет принадлежность к определенному типу задач. Ее длительность зависит от исполнителя. Но, тем не менее, оценим ориентировочное время ее выполнения dur_i. Кроме того, следует знать множество задач Pred(i), которые следует завершить перед началом выполнения задачи i. В процессе планирования и решения следует также знать статус задачи st(i) и ее приоритет pr(i). Таким образом, задачи можно описать так:

$$Task_i = Task_i(cl(i), dur(i), Pred(i), st(i), \text{pr}(i), t_{\text{нач}}(i)). \tag{2.7}$$

Опишем также исполнителя как некоторый объект, имеющий принадлежность к некоторому классу и имеющий определенную занятость:

$$Exec_j = Exec_j(cl(j), Sch(j)). \quad (2.8)$$

Определим ограничение на возможное время начала выполнения задачи i у специалиста j. Это ограничение будет сформировано, исходя из занятости исполнителя j и времени выполнения предшествующих работ для задачи i. Первое ограничение, связанное с занятостью исполнителя, будет иметь вид:

$$est1(i,j) = \min(t^*: Sch_j(t) = 0,\ t^* \le t \le t^* + dlit_{lj}^*) \quad (2.9)$$

Здесь $dlit_{lj}^*$ – оценка (математическое ожидание) случайной величины, описывающей длительность выполнения задачи i специалистом j. Формула (2.5) определяет время начала выполнения некоторой работы i специалистом j, исходя из занятости данного специалиста решением других задач: специалист должен быть свободен весь временной интервал, равный длительности выполнения им данной работы. Минимально возможное время начала такого интервала t^* и будет возможным началом выполнения данной работы у данного специалиста.

Ограничение, связанное с взаимной зависимостью задач, будет определяться по формуле:

$$est2(i,j) = \max_{l \in Pred(i)}\left(t_{\text{нач}}(l) + \sum_{j=1}^{M_k} x_{lj} \cdot dlit_{lj}^*\right). \quad (2.10)$$

Исходя из этого, можно определить нижнюю границу времени начала решения задачи i в случае, если она будет выполняться исполнителем j:

$$\begin{cases} t_{\text{нач}}(i,j) \geq est1(i,j) \\ t_{\text{нач}}(i,j) \geq est2(i,j) \end{cases}. \quad (2.11)$$

Критический путь проекта с учетом зависимости длительности решения задач от выполняющих их специалистов будет рассчитываться следующим образом:

$$T_{\text{крит}} = \max_l \left(t_{\text{нач}}(i) + \sum_{j=1}^{M_l} x_{lj} \cdot dlit_{lj}^{*} \right). \quad (2.12)$$

Формулы (1)-(12) представляют собой модель управления IT-проектом, отличающуюся учетом квалификации специалистов, их занятости, типизации задач.

Для решения оптимизационной задачи сформулируем целевые функции и ограничения. В качестве цели для задачи планирования выберем минимизацию общего времени выполнения всех задач:

$$T_{\text{крит}} \rightarrow min. \quad (2.13)$$

Задачу о назначениях на каждом этапе будем решать с точки зрения критериев Qual_1,.., Qual_L.:

$$\begin{cases} \text{Qual}_1 \rightarrow extr. \\ \quad \ldots \\ \text{Qual}_\text{L} \rightarrow extr. \end{cases} \quad (2.14)$$

Таким образом, сформулирована многокритериальная оптимизационная задача, позволяющая одновременно находить время начала каждой из задач, так и наилучшего исполнителя для нее. Особенность рассматриваемой оптимизационной задачи заключается в том, что задачу (2.13) необходимо решать одновременно с задачей (2.14).

2.2 Математическая модель оценки сроков выполнения IT-проектов и его отдельных задач

Одной из особенностей IT-проектов является возможность коррекции задач на самых разных стадиях его исполнения. Так, в результате тестирования в задаче могут быть обнаружены логические ошибки, связанные, например, с игнорированием каких-либо условий, неучетом важных факторов и т.д. Ошибки могут появиться в результате интеграции проекта или его части. Кроме того, коррекция задачи может возникнуть из-за изменившихся требований заказчика. Так, в результате очередного обсуждения или по завершении тестировании им некоторой версии реализованной задачи, у него могут возникнуть новые пожелания или сформироваться новые требования относительно данного функционала.

К сожалению, существующие методы (в первую очередь, PERT) в своих оценках могут учесть лишь наибольшее время выполнения той или иной операции. В данном случае необходим вероятностный механизм, который позволил бы учесть нюансы такой коррекции. Исследуемую задачу можно сформулировать следующим образом. Пусть имеется некоторый специалист i и задача j. Пусть оценено наименьшее (a_{ij}), наибольшее (b_{ij}) и наиболее вероятное (m_{ij}) время решения данной задачи. Предложить оценку времени выполнения задачи данным исполнителем в предположении, что она может подвергнуться коррекции.

Возьмем за основу оценку метода PERT, позволяющего оценить ожидаемую длительность по трем описанным выше оценкам [72]:

$$\xi_{ij} = \left(a_{ij}(t) + 4m_{ij}(t) + b_{ij}(t)\right)\Big/6. \qquad (2.15)$$

Предположим теперь, что задача j может быть скорректирована. На данном этапе будем оценивать вероятность коррекции данной задачи в результате обнаруженных на этапе тестирования ошибок. Следовательно,

данный вид коррекции связан не только с задачей, но и с исполнителем. Предположим, что некоторым образом удалось оценить вероятность коррекции данной задачи данным исполнителем. Это можно сделать, например, отнеся задачу к тому или иному классу и оценив по имеющимся статистическим данным вероятность появления ошибок у исполнителя i в решении задач такого класса. Возможно оценить вероятность ошибок в целом так или иначе, связав ее с уровнем исполнения специалиста i. Без ограничения общности, пусть p_{ij} – вероятность того, что исполнитель i допустит ошибку при решении задачи j. Кроме того, пусть каким-либо образом оценены вероятности наименьшего, наибольшего и наиболее вероятного времени коррекции задачи j исполнителем i. Обозначим их через переменные a_{1ij}, b_{1ij} и m_{1ij} соответственно. Если предположить, что случайную величину δ_{1ij}, описывающую время, затраченное на коррекцию, можно аппроксимировать распределением бета, то длительность коррекции можно оценить формулой, аналогичной (2.15):

$$\delta_{1ij} = \left(a_{1ij}(t) + 4m_{1ij}(t) + b_{1ij}(t)\right)\Big/_{6}. \qquad (2.16)$$

В общем случае есть ненулевая вероятность, что исполнитель i с первого раза не сможет скорректировать задачу j. Тогда можно предложить оценку длительности повторной коррекции. Для этого необходимо каким-либо образом оценить вероятность повторной коррекции и наименьшее, наибольшее и наиболее вероятное время выполнения такой коррекции.

В общем случае случайную величину, описывающую коррекцию выполнения исполнителем i задачи j, можно описать следующим образом:

$$\delta_{ij} = q_{1ij}\delta_{1ij} + q_{1ij}q_{2ij}\delta_{2ij} + \cdots \qquad (2.17)$$

Здесь:

ξ_{1ij}, ξ_{2ij}, ξ_{3ij}, … - случайные величины, описывающие соответствующую стадию выполнения работы; q_{1ij} – вероятность того, что исполнитель i

допустит ошибку при выполнении задачи j; q_{2ij} – вероятность того, что исполнителем i будет повторно допущена ошибка и т.д.

Оценка математического ожидания величины (2.17) в общем случае будет такая:

$$M^*\delta_{ij} = \sum_{l=0}^{\infty} slag_{ijl}, \quad (2.18)$$

где

$$slag_{ijl} = \prod_{k=0}^{l-1} q_{kij} \cdot \left(a_{kij}(t) + 4m_{kij}(t) + b_{kij}(t)\right)\Big/_{6}. \quad (2.19)$$

Однако, следует заметить, что оценки параметров для каждого последующего слагаемого в формуле (2.17) могут содержать все большую погрешность. Это происходит из-за сложности получения исторических данных на последующие коррекции и неточности моделей для их оценки. В связи с этим, для оценки δ_{ij} в формуле (2.17) можно оставить лишь первое (или несколько первых, если есть такая возможность) слагаемых.

Далее рассмотрим вопрос, связанный с возможным внесением изменений в техническое задание со стороны заказчика. Вероятность появления данной коррекции уже не будет связано с исполнителем, поскольку готовый функционал будет предоставлен только после его тщательного тестирования и проверки на соответствия с оговоренными функциями ранее. Пусть специфика задачи позволяет заранее оценить некоторую степень неопределенности и, как следствие, возможность внесения изменений со стороны заказчика. Пусть некоторым образом оценена вероятность pk_j данного события. Кроме того, следует некоторым образом оценить случайную величину η_{ij}, описывающую длительность коррекции. Если некоторым образом удалось оценить три временные характеристики: наименьшее (a_k_{ij}), наибольшее (b_k_{ij}) и наиболее вероятное (m_k_{ij}) время коррекции данной задачи j исполнителем i, то случайная величина η_{ij} может быть оценена аналогично формуле (2.14):

$$\eta_{ij}^{*} = \left(a\_k_{ij}(t) + 4m\_k_{ij}(t) + b\_k_{ij}(t)\right)\Big/_{6}. \quad (2.20)$$

К сожалению, повторная коррекция будет являться событием с крайне высокой степенью неопределенности, в связи с чем, оценивать соответствующие вероятности не представляется возможным.

Таким образом, длительность выполнения задачи j исполнителем i – это будет случайная величина, которая определяется следующим образом:

$$dlit_{ij} = \xi_{ij} + p_{ij} \cdot \delta_{ij} + pk_j \cdot \eta_{ij}. \quad (2.21)$$

Здесь ξ_{ij} – случайная величина, описывающая длительность выполнения задачи j специалистом i, которая определяется по формуле (2.15);

p_{ij} – вероятность коррекции выполнения задачи j исполнителем i;

δ_{ij} –случайная величина, описывающая коррекцию решения задачи j исполнителем i, которая определяется по формуле (2.17);

pk_j – вероятность изменения требований к задаче j заказчиком (или другим ЛПР);

η_{ij} - случайная величина, описывающая коррекцию решения задачи j исполнителем i в связи с изменившимися требованиями заказчика.

Рассмотрим подходы к оценке вероятностей, встречающихся в формуле (2.21). Данная величина будет зависеть от многих факторов. К ним, в частности, можно отнести:

- опыт исполнителя в решении задач подобного типа;

- квалификацию исполнителя;

- сложность задачи;

- качество входных данных, четкость требований в техническом задании

- и другие.

Следует отметить, что для подобной оценки важными могут оказаться и другие факторы, такие как уровень усталости исполнителя, его мотивация,

технические ограничения, сложность коммуникации для детализации задачи. Кроме того, существует ряд характеристик, специфических для IT-проектов, таких как частота изменений (рефакторинг), тип задачи (является ли данная задача новой или это рефакторинг или исправление ошибок) и другие. Выделим среди множества факторов те, которые, во-первых, являются значимыми, а, во-вторых, которые достаточно доступны (например, исторические данные, которые несложно получить из базы). К ним отнесем следующие (Таблица 2.1). В качестве шкалы выберем шкалу [0,1], где 1 будет нести максимальный риск от данного фактора, 0 – минимальный риск.

Таблица 2.1 - Факторы, влияющие на вероятность коррекции задачи

Фактор	Обозначение	Пояснение	Способ извлечения данных
Сложность задачи j	C_j	0 – самая простая задача; 1 – наиболее сложная	оценка от архитектора, от teamlead (или от исполнителей: покерное голосование)
Опыт исполнителя i в задачах типа j	E_{ij}	0 - максимальный опыт; 1 - полное отсутствие опыта	Автоматически (количество выполненных задач данного типа, количество правок по замечаниям и т.д.)
Квалификация исполнителя i	K_i	0 – исполнитель с высокой квалификацией; 1 – исполнитель с низкой квалификацией	Система грейдов (например, 0.2 – джуниор, 0.5 – middle и т.д.); Периодическая оценка от teamlead
Новизна задачи	N_j	0 – наличие абсолютно аналогичных задач; 1 - полное отсутствие аналогов;	Автоматический расчет, исходя из, например, анализа кодовой базы, анализа описания задачи
Качество ТЗ	F_j	0-высокое качество ТЗ; 1 низкое качество	Оценка от исполнителя, принимающего ТЗ

Сформулируем требования, которым должна удовлетворять математическая модель, оценивающая данную вероятность. В первую

очередь, эта модель должна гибко учитывать вклад каждого фактора в итоговый результат. Также желательно исключить граничные значения (в связи с тем, что невозможные и достоверные события крайне редко встречаются в реальных случаях, и в данной задаче практически невозможно гарантировать однозначное отсутствие коррекции после решения некоторой задачи, как и ее обязательное присутствие). С учетом того, что в Таблице 2.1 учтена лишь часть факторов, которые в целом, несложно измерить, целесообразно потребовать достаточно легкую расширяемость модели при возможности получения относительно точных количественных оценок новых факторов. Исходя из этого, остановимся на сигмоидальной функции, позволяющей не только снижать или повышать риск нелинейно, что в большей степени соответствует действительности, но и автоматически ограничивает диапазон в интервале (0,1), а также устойчива к нулевым и граничным значениям. Кроме того, данная функция «удобна» для добавления новых факторов в случае получения новых исторических данных. Общий вид данной формулы имеет вид:

$$P_{ij} = \frac{1}{1+e^{-\sigma}}, \tag{2.22}$$

где σ - аргумент, зависящий от комбинации факторов, приведенных в Таблице 2.1:

$$\sigma = \alpha_1 \cdot C_j + \alpha_2 \cdot E_{ij} + \alpha_3 \cdot K_i + \alpha_4 \cdot N_j + \alpha_5 \cdot F_j. \tag{2.23}$$

Здесь α_1, α_2, α_3, α_4, α_5 – коэффициенты, регулирующие вес того или иного фактора и его вклад в общую сумму. Эти параметры оцениваются, например, с помощью статистических методов. Формулы (2.22), (2.23) обладают очевидным преимуществом относительно простой мультипликативной формулы. Если некоторый фактор принимает нулевое значение (например, у исполнителя отсутствует опыт), то он просто не будет учтен в итоговой формуле. Кроме того, увеличение и снижение риска возникновения ошибки происходит плавно и нелинейно и не достигает предельных значений.

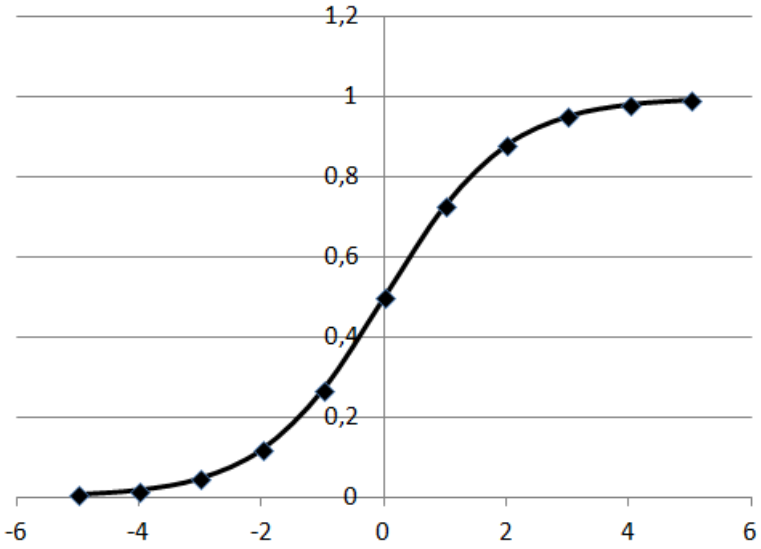


Рисунок 2.1 – График зависимости предложенной функции вероятности ошибки от аргумента

Аналогичный подход можно использовать для оценки вероятности коррекции задачи в результате ее демонстрации заказчику. Выделим основные факторы, которые будут влиять на эту вероятность и сгруппируем их в следующей таблице (Таблица 2.2).

Таблица 2.2 - Факторы, влияющие на вероятность коррекции задачи

Фактор	Обозначение	Пояснение	Способ извлечения данных
Сложность задачи j	C_j	0 – самая простая задача; 1 – наиболее сложная	Оценка от архитектора, от teamlead (или от исполнителей: покерное голосование)
Четкость требований заказчика	D	0-требования полностью ясны; 1 – низкое качество, требования размыты	Оценка от исполнителя, принимающего ТЗ
Тип задачи	Tp_j	0-задачи с минимальным риском коррекции (например, DevOps, развертывание CI/CD), 1 – задачи с максимальным риском (например, frontend-разработка, дизайн и т.д.)	Классификация задач и автоматическое отнесение к соответствующему типу
Уровень вовлеченности заказчика	LI	0 – заказчик максимально вовлечен; 1 – отсутствие вовлеченности	Оценивается лицом (лицами), общающимся с заказчиком.
Предыдущий опыт взаимодействия с этим заказчиком	PE	0– заказчик ясно излагает свои требования и редко потом вносит правки; 1– заказчик склонен вносить много правок	Оценка по историческим данным
Наличие этапа предварительного согласование	PA	0 – присутствие предварительного солгасование, обсуждение mockup'ов, прототипов; 1 – отсутствие предварительного солгасования	Оценивается лицом (лицами), общающимся с заказчиком.

С учетом наличия данных факторов, формула pk_j будет аналогична формуле (2.20), где σ будет определяться формулой:

$$\sigma = \beta_1 \cdot C_j + \beta_2 \cdot D + \beta_3 \cdot Tp_j + \beta_4 \cdot LI + \beta_5 \cdot PE + \beta_6 \cdot PA. \quad (2.24)$$

Как и в формуле (2.20), здесь коэффициенты β_1, β_2, β_3, β_4, β_5, β_6 – регулируют вес соответствующего фактора.

Таким образом, разработана математическая модель оценки сроков выполнения отдельных задач, отличающаяся учетом коррекции задач, обеспечивающая повышенную точность по сравнению с аналогами.

Формула (2.21) отличается повышенной точностью по сравнению с аналогами (в частности, с формулой (2.15)), поскольку учитывает не только специфику каждого исполнителя, но и коррекцию задач. Однако, при составлении план-графика, когда для задач предварительно рассчитывается резерв, заранее неизвестно, какой исполнитель будет выполнять отдельные задачи. Это связано с тем, что резерв рассчитывается, в частности, по позднему времени начала работ, которые рассчитываются, начиная с самого последнего события. На этом этапе задача о назначениях еще не решена. Поэтому возникает необходимость получения более точной оценки длительности отдельных задач без наличия информации о специалистах.

В ряде случаев для определенных задач можно получить некоторую оценку вероятности назначения ее тому или иному исполнителю. Например, в случае возникновения наиболее сложной задачи целесообразно отдать ее более опытному специалисту. Если некоторый исполнитель занимался решением подобных задач, то вероятность качественного исполнения им данной задачи будет выше, чем у неопытного специалиста, который никогда с такой спецификой не сталкивался. Исходя из этого, для ряда задач, как было отмечено выше, можно получить распределение вероятность выполнения их теми или иными исполнителями. Пусть p_choise_{ij} – вероятность безошибочного выполнения задачи j исполнителем i. Тогда, оценив предварительно ожидаемые длительности задач исполнителями $dlit_{ij}$ по формуле (2.15), общую длительность задачи i можно оценить с помощью формулы:

$$dur_j = \sum_{i=1}^{M} dlit_{ij} \cdot p\_choise_{ij}. \quad (2.25)$$

В ряде случаев, когда набрана абсолютно новая команда исполнителей, и сложность новых задач оценить крайне затруднительно, подобные вероятности могут отсутствовать. В этом случае предполагается, что значения a_{ij}, b_{ij} и m_{ij}, как правило, предлагаются самими исполнителями в качестве оценок. Тогда длительность можно оценить по средним значениям:

$$dur_j = \frac{\sum_{i=1}^{M} dlit_{ij}}{M}. \quad (2.26)$$

Таким образом, предложена математическая модель, оценки сроков выполнения как отдельных задач, так и всего проекта в целом, отличающаяся учетом коррекции задач, обеспечивающая повышенную точность по сравнению с аналогами.

2.3 Имитационная модель процесса выполнения IT-проекта

Для оценки сроков выполнения всего проекта использование метода PERT, как показал опыт, как правило, приводит к погрешностям. Это связано, в первую очередь, с наличием неопределенностей в данной задаче. Помимо случайного характера длительности обслуживания эта неопределенность заключается в возможной коррекции как всего проекта так и его составляющих из-за разнообразных причин, возникающих как со стороны заказчика, так и со стороны исполнителей. Второй фактор, влияющий на длительность, заключается в необходимости учета взаимной зависимости задач. Большое число предшествующих задач увеличивают вероятность запаздывания начала данной задачи из-за вероятностного характера длительностей. В частности, многочисленные экспериментальные исследования показали, что даже в классической своей постановке метода PERT длительность всего проекта оказывается выше, чем имеющиеся оценки. В связи с этим, для оценки сроков проекта в условиях

неопределенности и коррекции отдельных задач целесообразно использование аппарата имитационного моделирования.

Средой моделирования выбрана AnyLogic, так как она позволяет реализовать потоковые и дискретные процессы, объединять задачи, учитывать распределения длительности и интегрироваться с базой данных. Для оценки длительности проекта средствами имитационного моделирования необходимо решить следующие задачи:

- описать случайную длительность решения каждой из задач проекта;
- имитировать процесс решения задач за данную длительность;
- закрепление за задачами исполнителей;
- реализовать возможность коррекции отдельных задач;
- учесть взаимную зависимость при решении задач;
- организовать сбор необходимых статистических данных средствами AnyLogic.

Рассмотрим решение каждой из вышеперечисленных задач более подробно. Одна из проблем, которая возникает при моделировании случайных величин, описывающих длительность решения отдельных задач, связана с имитацией ее выполнения в течении заданного времени. Как было отмечено ранее, закон бета является достаточно универсальным для описания таких длительностей. Бело предположено, что для каждой задачи каждого исполнителя известны:

- наименьшее время, затраченное на решение данной задачи (a_{ij});
- наибольшее время, затраченное на решение данной задачи (a_{ij});
- наиболее вероятное время m_{ij}.

На выходе необходимо получить случайную величину, распределенную по закону бета с параметрами (a_{ij}, b_{ij}), описывающие границы распределения данной случайной величины, а также параметры (p_{ij}, q_{ij}) данного распределения. С учетом того, что параметры a_{ij} и b_{ij} будем считать известными, возникает задача оценки неизвестных параметров

распределения (p_{ij}, q_{ij}). Для этого воспользуемся известными формулами математического ожидания и дисперсии распределения бета:

$$\begin{cases} M\xi_{ij} = a_{ij} + \left(b_{ij} - a_{ij}\right) \cdot p_{ij} \Big/ \left(p_{ij} + q_{ij}\right) \\ D\xi_{ij} = \left(b_{ij} - a_{ij}\right)^2 \Big/ \left(\left(p_{ij} + q_{ij}\right)^2 \cdot \left(p_{ij} + q_{ij} + 1\right)\right) \end{cases} \quad (2.27)$$

Решив данную систему уравнений относительно неизвестных (p_{ij}, q_{ij}), будем иметь:

$$\begin{cases} p = \left((b - M\xi) \cdot (M\xi - a)^2 - D\xi \cdot \left((M\xi - a)\right)\right) \Big/ D\xi \cdot \left((b - a)\right) \\ q = \left((b - M\xi)^2 \cdot (M\xi - a) - D\xi \cdot \left((b - M\xi)\right)\right) \Big/ D\xi \cdot \left((b - a)\right) \end{cases} \quad (2.28)$$

Таким образом, длительность выполнения задач в имитационной модели будет моделироваться следующим образом.

Шаг 1. Неизвестный параметр Mξ оценивается по формуле (2.15).

Шаг 2. Дисперсия оценивается по правилам трех сигм:

$$D\xi_{ij} = \left(b_{ij} - a_{ij}\right)^2 \Big/ 6. \quad (2.29)$$

Шаг 3. Длительность будет задана законом бета, параметра a и b которого будут взяты из базы данных, а параметры p и q – оценены по формуле (2.28).

Сами параметры находятся с помощью динамических переменных и связей системной динамики (Рисунок 2.2).

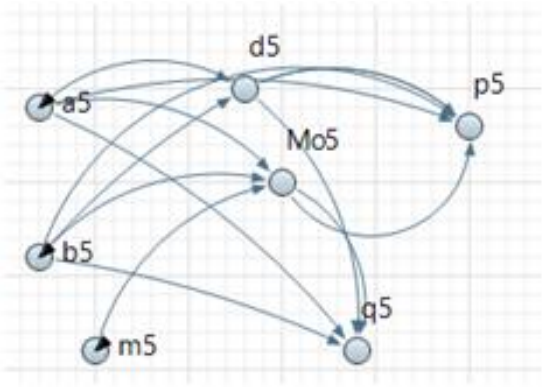


Рисунок 2.2 - Фрагмент имитационной модели для определения длительностей задач

Имитировать процесс выполнения некоторой задачи будем средствами объекта Service, каждый из которых имитирует решение задачи исполнителем, процесс выбора которого осуществляется средствами ResourcePool. Каждый исполнитель является ресурсом, который имеет уникальные параметры исполнения задачи, описываемой объектом Service. Соответствующее время задано библиотечной функцией beta() с параметрами, определяемыми формулами (2.25) и рисунок 2.2.

Далее смоделируем сам проект как множество взаимосвязанных объектов Service. Моделирование взаимной зависимости между задачами требует организации связи между объектами Service. Если некоторое событие является начальным для нескольких работ, оно моделируется блоком Split. Если для выполнения работы необходимо завершения нескольких работ, между ними ставится блок assembler. Коррекция некоторых работ добавляется с помощью условного блока типа SelectOutput, где с некоторой вероятностью заявка завершает работу, а с оставшейся требует повторного выполнения. Если при коррекции необходимо изменить параметры бета-распределения (что соответствует ситуации, когда время на доработку, например, меньше, чем время на решение задачи), то соответствующий код, задающий зависимость изменения параметров (функционально или напрямую через изменение предыдущих значений)

пишется в действиях при выходе false из порта SelectOutput. Для построения гистограммы в модель добавлен объект chart, данные в который заносятся при завершении очередного прогона модели.

Далее смоделируем сам проект как множество взаимосвязанных объектов Service, каждый из которых имитирует решение задачи за время, заданное функцией beta() с параметрами, нахождение которых описывалось выше. Если некоторое событие является начальным для нескольких работ, оно моделируется блоком Split. Если для выполнения работы необходимо завершения нескольких работ, между ними ставится блок assembler. Коррекция некоторых работ добавляется с помощью условного блока типа SelectOutput, где с некоторой вероятностью заявка завершает работу, а с оставшейся требует повторного выполнения. Если при коррекции необходимо изменить параметры бета-распределения (что соответствует ситуации, когда время на доработку, например, меньше, чем время на решение задачи), то соответствующий код, задающий зависимость изменения параметров (функционально или напрямую через изменение предыдущих значений) пишется в действиях при выходе false из порта SelectOutput. Для построения гистограммы в модель добавлен объект chart, данные в который заносятся при завершении очередного прогона модели.

Предварительно анализировалось поведение случайной величины, имитирующей решение отдельной задачи при условии, что она была скорректирована (один или несколько раз). Как было упомянуто ранее, соответствующая случайная величина в отсутствии коррекции достаточно точно оценивается распределением бета. Однако, повторная доработка (пусть и за меньшее время) существенно изменяет вид исследуемой случайной величины.

Рассмотрим теоретические и эмпирические результаты исследования случайной величины, описывающей длительности выполнения некоторой работы с возможной коррекцией.

Вначале рассмотрим более простой вариант обслуживания:

- вероятность от опыта к опыту не меняется;

- параметры распределения времени обслуживания от опыта к опыту не меняются.

В этом случае математическое ожидание описывается формулой (2.5).

Без ограничения общности, выберем следующие параметры распределения бета:

a=10; b=15; p=2; q=4.

На Рисунок 2.3 приведена гистограмма для случая, если вероятность успешного выполнения работы равна 0.8.

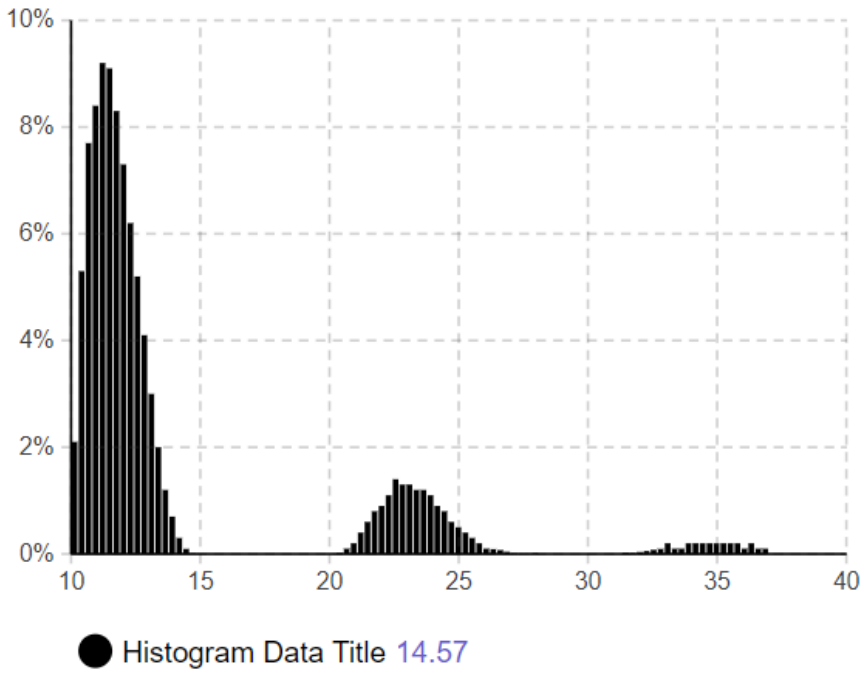


Рисунок 2.3 - Результаты для p=0.9

Как видно из данного рисунка, в целом случайная величина ведет себя достаточно предсказуемо. С вероятностью 0.8 она имеет обычное распределение бета в интервале [10, 15]. С вероятностью 0.2*0.8=0.16 величина распределена в интервале [20,30]. Далее совсем невелик процент того, что работа должна быть еще раз выполнена (и, как следствие, случайная величина будет распределена в интервале [30, 45]) и т.д.

В Таблице 2.3 приведены сравнительные величины математических ожиданий длительности выполнения работ и ее оценки с помощью аппарата математического моделирования для разных значений вероятностей выполнения работ.

Таблица 2.3 - Сравнительные величины математических ожиданий длительности выполнения работ и ее оценки

№	p	$M\xi$	$M\xi^*$	A	A*	B	B*
1	0.5	23.33	23.37	10.004	-26,05	167.325	72,787
2	0.6	19.44	19.38	10.011	-17,41	144.008	56,177
3	0.7	16.67	16.43	10.011	-10,69	111.54	43,976
4	0.8	14.58	14.56	10.008	-5,098	80.033	34,214
5	0.9	12.96	12.97	10.008	0,389	71.559	25,541

Сравнив результаты, видим, что в среднем погрешность оценки математического ожидания с помощью аппарата имитационного моделирования относительно невелика. Однако, оценка размаха случайной величины, полученная по правилам трех сигм, абсолютно неприменима.

Далее будем менять длительность повторных коррекций. Уменьшим, например, размах случайной величины, отвечающей за каждую последующую повторную коррекцию, на 10%. Получим следующую гистограмму длительности выполнения отдельной задачи (Рисунок 2.4).

Таким образом, анализируя поведение случайной величины, описывающей длительность выполнения некоторой работы при условии ее возможной коррекции, можно прийти к следующим выводам.

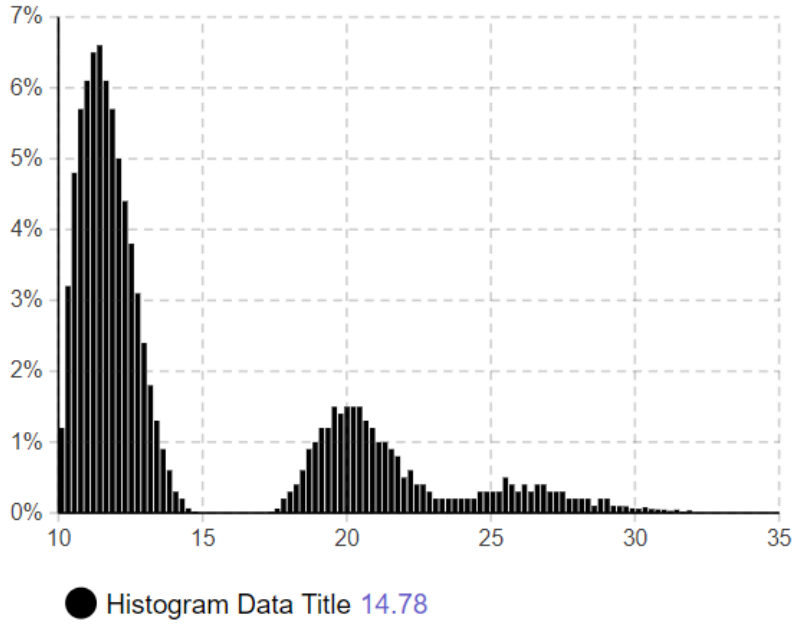


Рисунок 2.4 - Результаты для модели с изменяемыми параметрами

1. Закон распределения случайной величины многомодальный. В случае, если характеристики длительностей принимают такие значения, при которых оценку плотности можно представить в виде совокупности отдельных графиков, соответствующих бета-распределению, оценка вероятности завершения проекта в некоторый временной интервал будет определяться несколько проще, чем в случае «наложения» соответствующих волн.

2. Математическое ожидание случайной величины увеличивается незначительно. При этом существенно меняется размах случайной величины, ее правая граница.

Следующим шагом будет являться исследование случайной величины, отвечающей за длительность всего проекта в условиях неопределенности, связанной с разной длительностью выполнения разными специалистами своих задач, а также возможной коррекцией как всего проекта, так и его отдельных составляющих.

Для этого было проведено множество экспериментов, позволяющих в заданных условиях получить всю необходимую информацию о длительности

проекта. Целью разработки экспериментов является, в первую очередь, исследование (и оценка) его временных характеристик проекта при закреплении задач за заданными исполнителями с заданным уровнем неопределенности. Кроме того, имитационная модель использовалась для проведения экспериментов с целью подтверждения эффективности разработанных алгоритмов планирования.

Анализировались следующие гипотезы.

Гипотеза 1. Может ли ожидаемая длительность проекта быть спрогнозирована методом PERT:

$$M(proj) = \sum_{i \in Krit} M\xi_i. \quad (2.30)$$

Гипотеза 2. Может ли дисперсия проекта быть спрогнозирована методом PERT:

$$D(proj) = \sum_{i \in Krit} D\xi_i. \quad (2.31)$$

Гипотеза 3. Можно ли размах распределения оценить с помощью правила трех сигм:

$$\begin{cases} min = M(proj) - 3\sigma \\ max = M(proj) + 3\sigma \end{cases} \quad (2.32)$$

Гипотеза 4. Можно ли распределение случайной величины, описывающей длительность проекта, аппроксимировать нормальным распределением.

Опишем эксперимент более подробно. На вход модели поступает проект с закрепленными исполнителями за каждой задачей. На выходе получим статистику о времени его выполнения, а также расчет интересующих вероятностей. В первую очередь, интерес представляет вероятность завершения проекта за критическое время, определенное формулой (2.30). Кроме того, в модели задан поиск вероятности завершения к максимальному сроку, рассчитанному с помощью второй формулы в системе (2.32).

Опишем эксперименты более подробно. На первом этапе использовался небольшой проект, состоящий из 16 задач, взаимосвязь которых представлена на Рисунке 2.5.

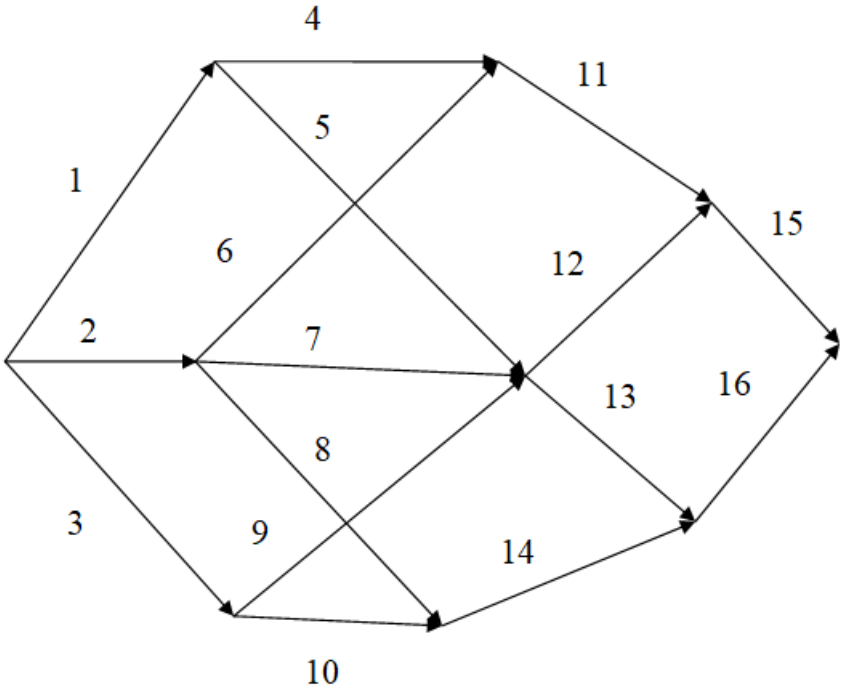


Рисунок 2.5 - Сетевой график проекта

Пусть, без ограничения общности, имеются 4 исполнителя, длительности выполнения задач которыми представлены в Таблице 2.4.

Таблица 2.4 - Оценки длительностей выполнения каждой из задач каждым исполнителем

Задача	Исполнит. 1				Исполнит. 2				Исполнит. 3				Исполнит. 4			
	a	m	b	M*	a	m	b	M*	a	m	b	M*	a	m	b	M*
1	4	7	8	6,667	6	8	10	8,000	5	8	9	7,667	4	6	7	5,833
2	6	9	10	8,667	4	7	9	6,833	5	8	9	7,667	5	7	9	7,000
3	8	9	12	9,333	5	7	10	7,167	9	10	11	10,00	7	9	11	9,000
4	5	7	9	7,000	7	9	12	9,167	6	8	9	7,833	5	7	9	7,000
5	6	8	10	8,000	6	9	11	8,833	7	9	11	9,000	6	8	9	7,833
6	5	7	10	7,167	7	9	11	9,000	6	8	10	8,000	5	7	10	7,167
7	7	10	12	9,833	5	8	10	7,833	7	9	11	9,000	7	8	9	8,000
8	8	10	12	10,00	6	8	10	8,000	7	9	12	9,167	6	8	9	7,833
9	3	5	8	5,167	5	6	7	6,000	6	8	10	8,000	3	5	6	4,833
10	4	6	7	5,833	5	7	8	6,833	5	7	8	6,833	4	5	7	5,167
11	5	8	10	7,833	5	7	8	6,833	6	7	9	7,167	4	6	8	6,000
12	6	8	10	8,000	6	7	8	7,000	4	7	10	7,000	5	7	9	7,000
13	7	9	11	9,000	5	8	10	7,833	6	8	11	8,167	6	9	10	8,667
14	5	7	10	7,167	7	9	11	9,000	7	8	10	8,167	5	8	9	7,667
15	4	6	9	6,167	6	8	10	8,000	6	8	9	7,833	4	6	8	6,000
16	5	8	10	7,833	7	9	11	9,000	8	10	11	9,833	5	7	9	7,000

Зададим длительности выполнения каждой задачи каждым исполнителем в соответствии со значениями из Таблицы 2.4, как представлено на Рисунке 2.2. Получим модель, основной фрагмент которой представлен на Рисунке 2.6.

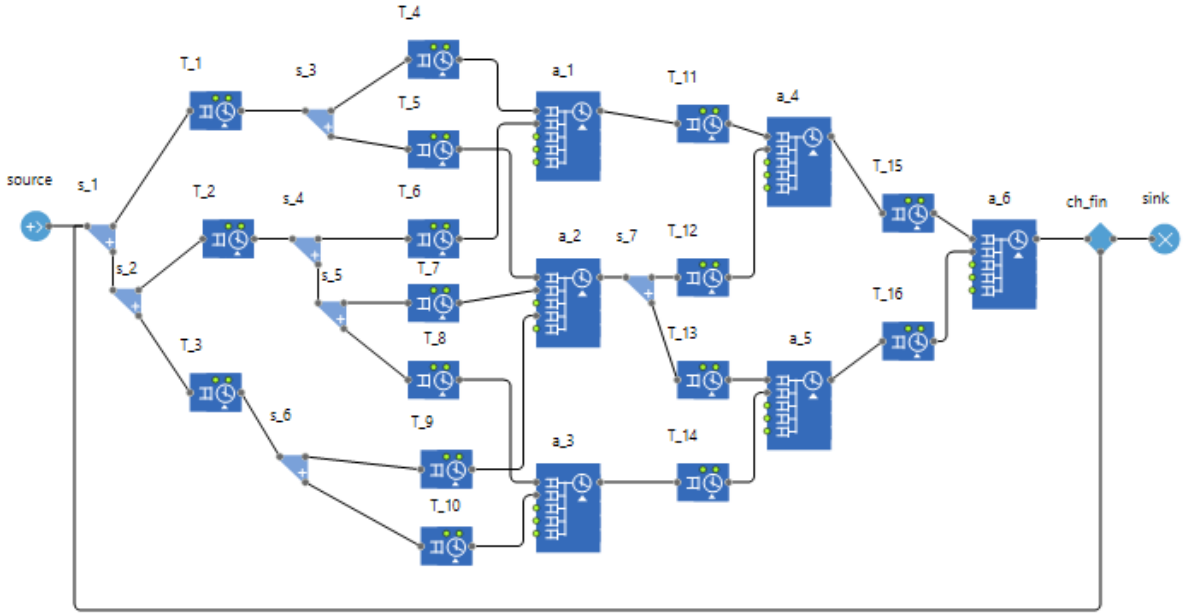


Рисунок 2.6 - Основной фрагмент модели

Будем менять назначения исполнителей и анализировать временные показатели проекта. После серии подобных экспериментов можно оценить как наилучшее распределение исполнителей, так и усредненные значения временных характеристик проекта.

Рассмотрим результаты одного из приведенных экспериментов. Соответствие между исполнителями и идентификаторами задач приведено в Таблице 2.5.

Таблица 2.5 - Соответствие между исполнителями и задачами

Задача (номер)	Исполнитель	Задача (номер)	Исполнитель
1	4	9	4
2	2	10	2
3	3	11	4
4	2	12	2
5	1	13	3
6	1	14	1
7	2	15	1
8	3	16	4

Как видно из данной таблицы, задачи распределены между исполнителями следующим образом. Два исполнителя (под номерами 2 и 4) должны выполнить в проекте 5 задач, а исполнители 1 и 3 – по 3 задачи. Таким образом, присутствует небольшая неравномерность при назначении исполнителей. Гистограмма случайной величины, описывающей длительность проекта, представлена на рис. 2.7. При использовании классических формул PERT и CPM, на критическом пути при данном закреплении оказываются задачи с идентификаторами 3, 10, 14 и 16. Ожидаемая длительность согласно методу PERT равна:

$$M\xi^* = 10 + 6.833 + 7.167 + 7 = 31.$$

Как можно видеть, расхождение с реальными данными составляет около 30%. Такая ошибка возникла из-за того, что метод PERT учитывает лишь взаимную зависимость задач. В частности, задачи 4,7 и 10, исходя из специфики сетевого графика на Рисунке 2.5, можно считать параллельными и, следовательно, их условно можно выполнять параллельно. Однако, с учетом того, что на каждую из них назначен исполнитель 2, каждая из них будет выполнена с учетом взаимной зависимости.

Иными словами, для использования оценок длительности проекта, необходимо учитывать не только взаимную зависимость задач, но и ограничение на работу исполнителей, описанное формулами (2.7) и (2.9) (второе неравенство). В противном случае, как видно из данного примера, существующие оценки будут неприемлемы.

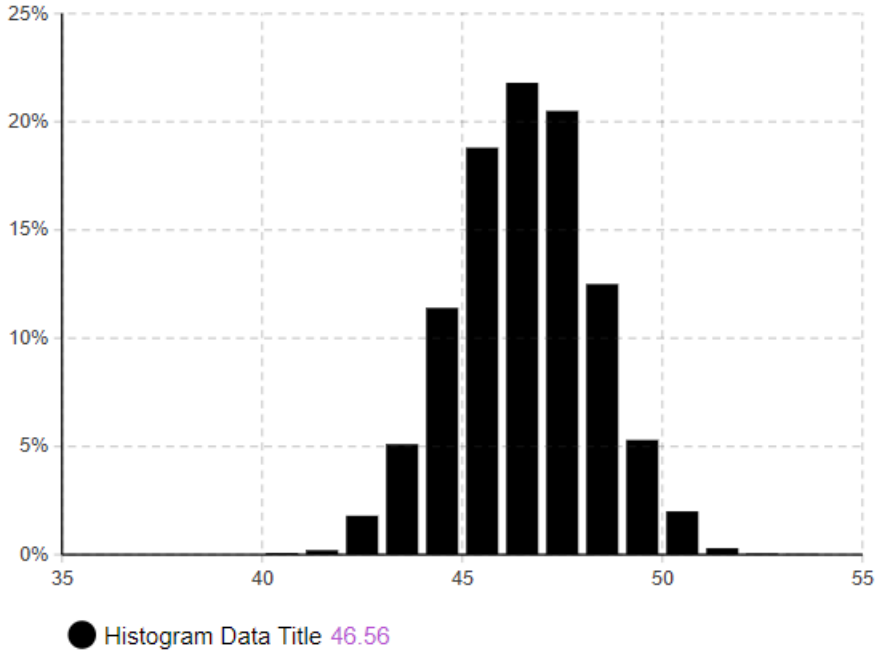


Рисунок 2.7 - Гистограмма длительности проекта при распределении задач согласно таблице 2.5

Таким образом, первую гипотезу в рамках исследуемой задачи можно отклонить. С учетом данного факта можно отклонить и третью гипотезу, поскольку она зависит от первой. Опыты показывают, что в отсутствии коррекции отдельных задач погрешности при оценке дисперсии формулами (2.28) будут относительно небольшими. Поэтому условно вторую гипотезу можно принять.

Изменим теперь в Таблице 2.5 назначение одной задачи 4. Закрепим ее за исполнителем 4. Получим следующую гистограмму распределения длительности обслуживания (Рисунок 2.8).

Как видно из данного рис., более эффективное назначение исполнителей уменьшило общую длительность выполнения проекта более, чем на 11%.

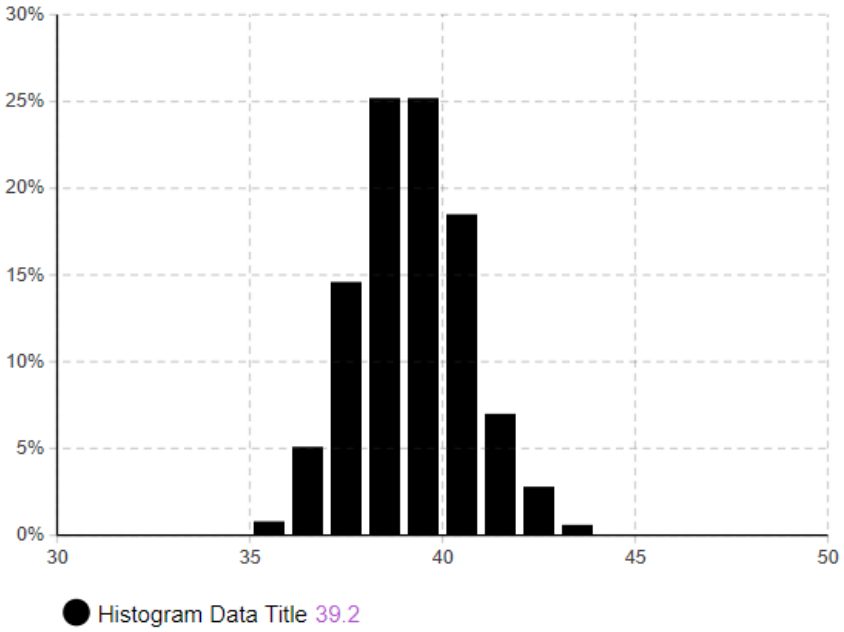


Рисунок 2.8 - Гистограмма длительности проекта при скорректированном распределении задач

Далее рассмотрим возможность коррекции задач. Допустим возможность ошибки в решении лишь одной задачи, не стоящей на критическом пути (это задача 2, вероятность ошибки 10%). Скорректированная гистограмма будет иметь следующий вид (Рисунок 2.9).

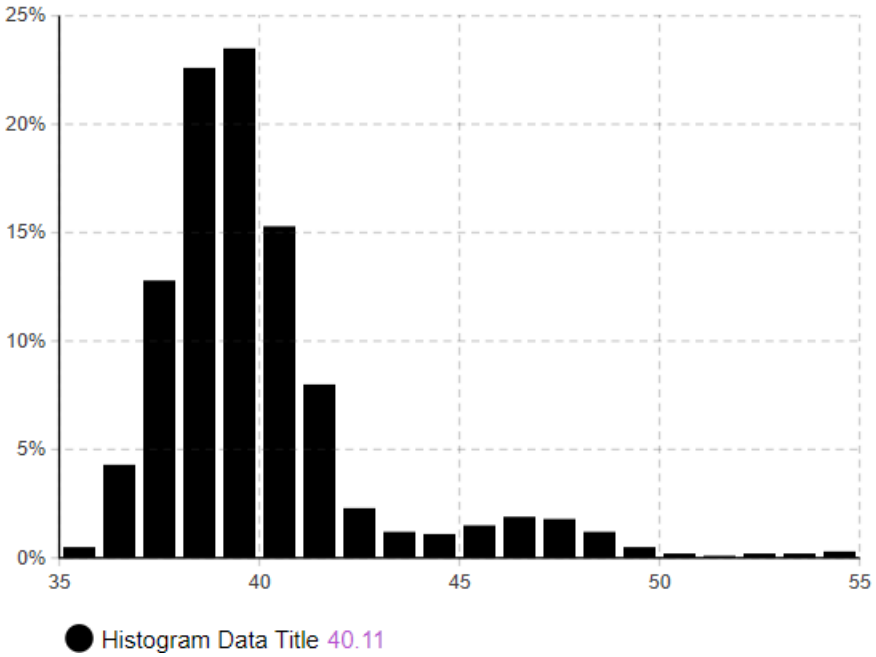


Рисунок 2.9 - Гистограмма после коррекции одной некритической задачи

Добавим также в модель возможность расчета разнообразных вероятностных характеристик по статистическим данным. В частности, будем рассчитывать вероятности следующих событий:

- проект завершится в момент, предсказанный с помощью метода PERT;

- проект завершится в самый поздний из моментов, предсказанным методом PERT.

Представим результаты еще одного эксперимента для проекта, состоящего из 20 задач. Без ограничения общности, для данного проекта проведем исследование лишь влияния коррекции.

В связи с этим, зададим сразу каждую задачу на сетевом графике ребром с весом, который будет определять идентификатор задачи и ожидаемую длительность, рассчитанную по наименьшему, наибольшему и наиболее вероятному значению с помощью PERT. Данный сетевой график представлен на Рисунке 2.10 (жирными линиями выделен критический путь).

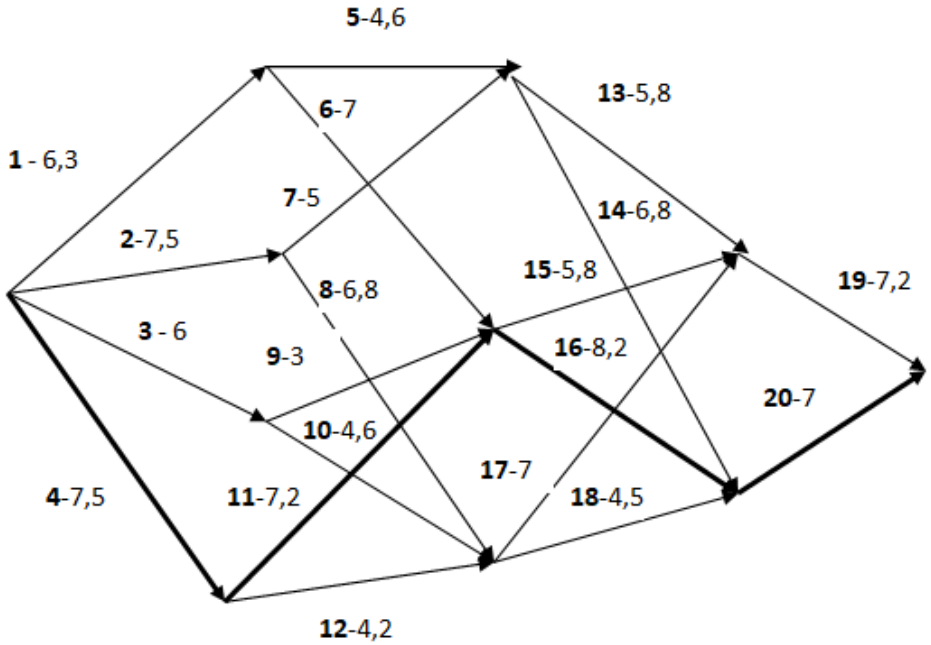


Рисунок 2.10 - Сетевой график проекта

По данному графику создадим имитационную модель, для каждой задачи которой предусмотрим коррекцию с помощью условных блоков, обозначенных на схеме идентификаторами SO1,…,SO20. Она представлена на Рисунке 2.11.

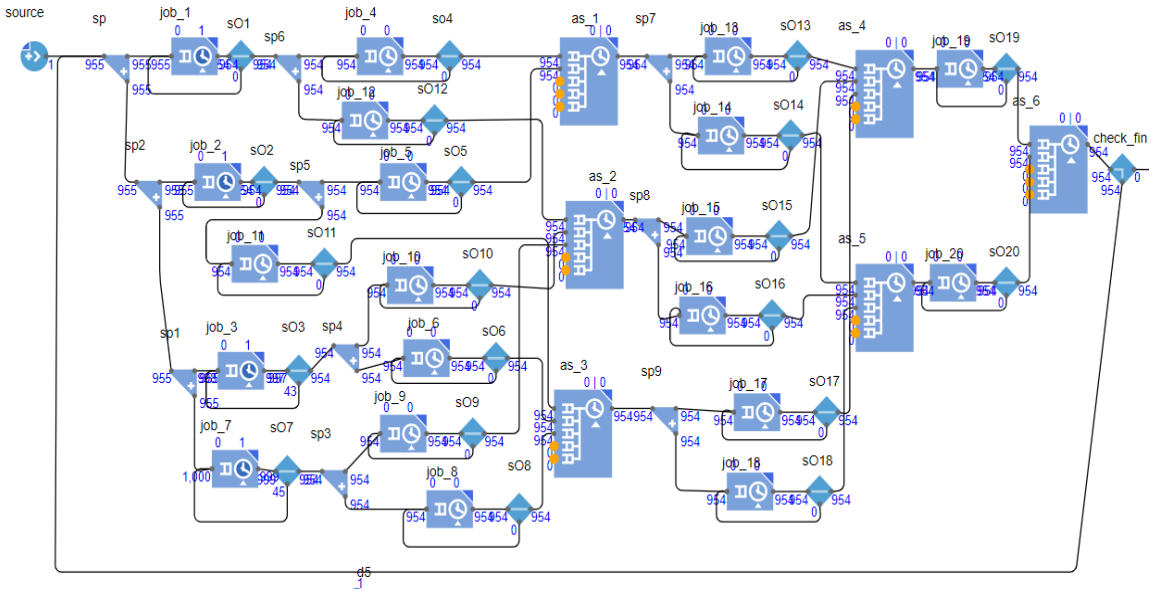


Рисунок 2.11 - Основной фрагмент модели

В теле каждого условия записывается изменение вероятности коррекции (закон, по которому задача будет (или не будет) скорректирована в следующий раз) и изменение размаха и параметров случайной величины, описывающей повторное исполнение задачи.

Будем последовательно увеличивать число задач, которые могут быть скорректированы. Результаты представлены в приложении В. Опишем их кратко в следующей таблице.

Таблица 2.6. - Результаты эксперимента

	$M\eta$	$\sqrt{D\eta}$	мин	макс	P(X< $M\eta^*$)	$P(X < M\eta^* + 3\sigma)$
все задачи без коррекции	32.626	1.52	27.525	38.507	0.031	0.955
0.05 (1 задача на кр.пути)	32.949	2.141	28.006	49.19	0.022	0.917
0.05 (2 з. на кр.пути)	33.27	2.619	27.3	49.293	0.025	0.871
0.05 (1 з. на к.п. и 1 з. не на к.п.)	33.075	2.263	27.755	49.844	0.024	0.902
Четверть задач с коррекцией	33.911	3.208	27.453	54.833	0.024	0.783
Половина задач с коррекцией с вер. 0.05	34.436	3.422	27.453	58.216	0.017	0.705
Все с коррекцией 0.05	36.015	4.359	27.794	58.062	0.012	0.532
Все с корр. 0.05, одна з. на к.п. с корр. 0.2	37.201	5.388	27.52	73.661	0.08	0.446

Приведем примеры нескольких гистограмм (все гистограммы приведены в приложении В). На Рисунке 2.12 приведены результаты для случая, если коррекции с вероятностью 0.05 может быть подвергнута одна задача, не лежащая на критическом пути, и одна задача с критического пути.

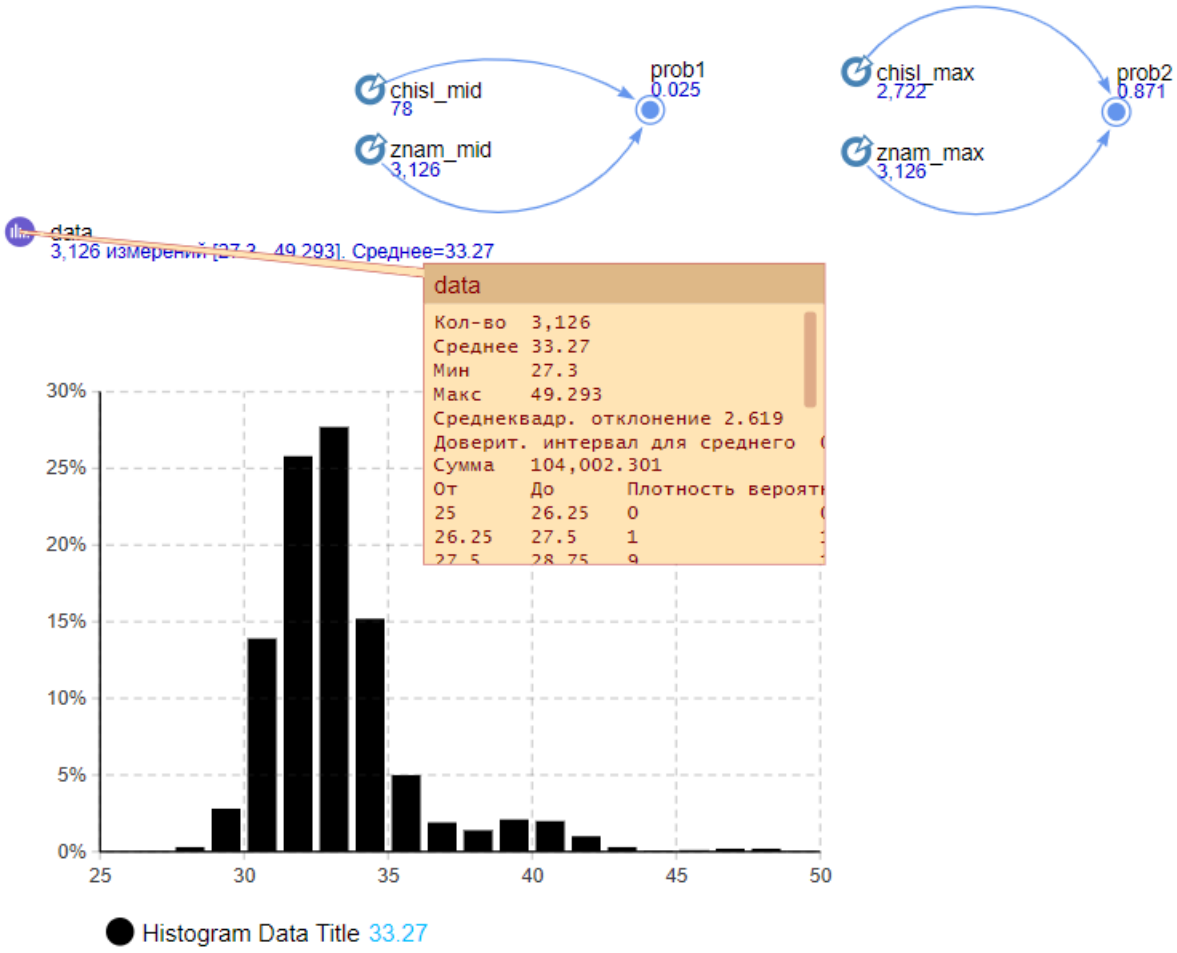


Рисунок 2.12 - Гистограмма при коррекции двух задач

Проанализируем полученные результаты. Как видно из Рисунка 2.12, вероятность того, что проект завершится к планируемому с помощью PERT сроку, составляет 0.025. Такая маленькая вероятность получена по двум причинам. Первая изложена ранее и заключается в том, что метод не учитывает занятость исполнителей, закрепленных за задачами. Вторая причина (и «хвост» распределения) имеет место из-за коррекции. Очевидно, что чем больше задач потенциально могут быть скорректированы, тем менее прогнозируемы искомые величины и более «тяжел» «хвост» распределения. В частности, если допустить, что любая задача может быть скорректирована с вероятностью 0.05, а одна задача – с вероятностью 0.2, то получим следующую картину (Рисунок 2.13).

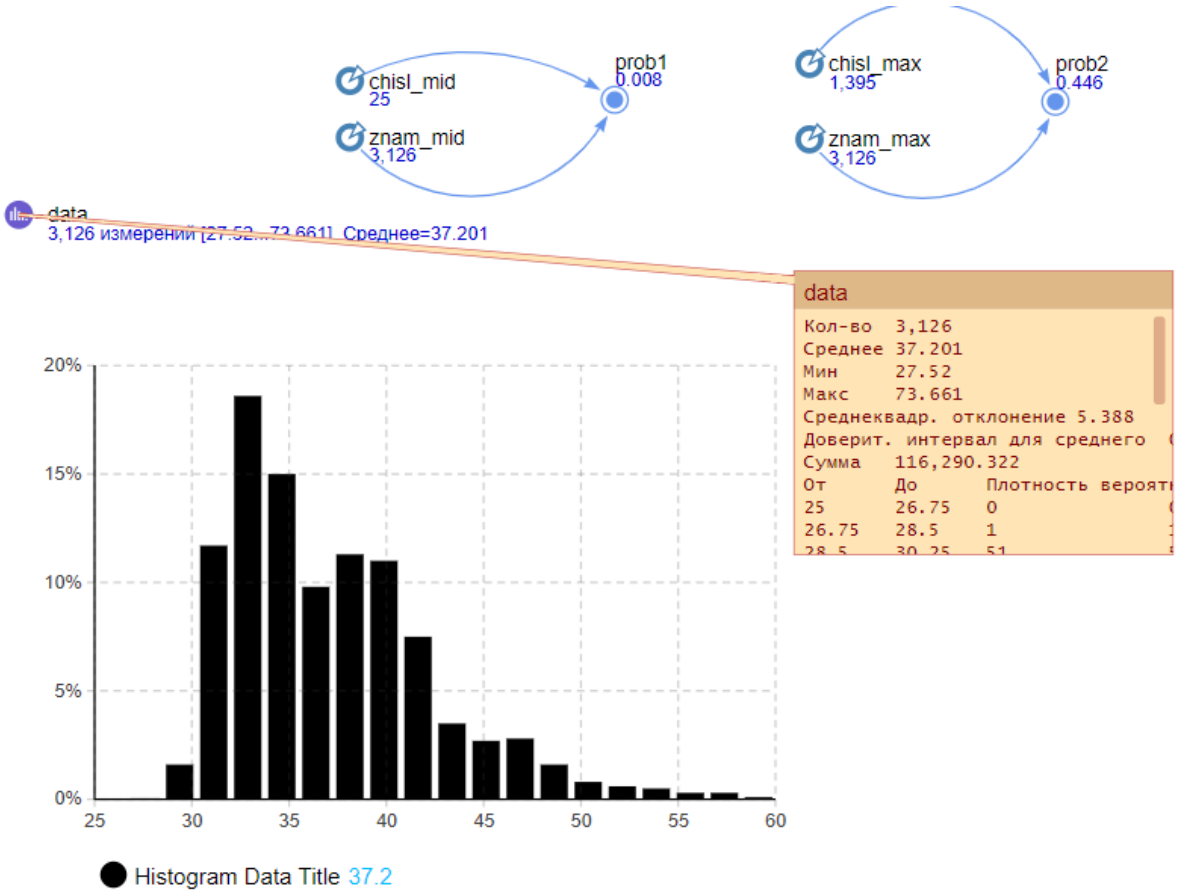


Рисунок 2.13 - Гистограмма при возможной коррекции всех задач

Как видно из этих результатов, даже небольшая вероятность коррекции задач сказывается на сроках всего проекта. Особенно чувствительно это влияние при коррекции задач, стоящих на критическом пути. Кроме того, длительность проекта существенно зависит от графика исполнителей (т.е. от второго неравенства, представленного в (2.9)). Все это говорит о том, что при большом числе неопределенностей временные характеристики проекта лучше всего получить с помощью разработанной имитационной модели, которая учитывает все эти нюансы.

Исходя из множества проведенных экспериментов можно сделать следующие выводы под выдвинутым ранее гипотезам.

В результате проведения множества экспериментов были получены следующие выводы.

Вывод 1. В случае отсутствия коррекции ожидаемая длительность может быть оценена по формуле (2.30) с определенной погрешностью (около

10%). Величина погрешности зависит от числа задач проекта, их временных характеристик (в первую очередь, от размаха) и от числа критических путей. Если задачи проекта допускают коррекцию, то величина погрешности увеличивается в несколько раз (в зависимости от вероятности такой коррекции). В этом случае формула (2.30) может быть применена только в случае, если оценка длительностей отдельных задач была проведена по формулам (2.21).

Вывод 2. В случае отсутствия коррекции дисперсию и размах в целом можно оценить формулами (2.31) и (2.32). При появлении коррекции дисперсия и верхняя граница распределения существенно возрастают.

Вывод 3. Без отсутствия коррекции с определенной погрешностью длительность проекта можно оценить нормальным распределение, хотя лучше для такой оценки подходит закон бета. В случае коррекции длительности задач на выходе получается, как правило, многомодовое распределение с явно выраженной правосторонней асимметрией.

Таким образом, была получена модель оценки длительности проекта, отличающаяся учетом коррекции задач, обеспечивающая повышенную точность по сравнению с аналогами. Данная модель получена с помощью системы имитационного моделирования AnyLogic.

2.4 Выводы

В данной работе процесс управления организацией деятельности в IT-компании был сведен к рассмотрению проблематики распределения задач по исполнителям и планированию времени начала решения каждой из отдельных задач.

1. Анализ специфики управления организационной деятельностью в IT-компании позволил выделить квалификацию исполнителей как важный нюанс при формировании модели управления IT-проектом. Кроме того, была учтена занятость исполнителей решением других задач, а также ограничения,

связанные с взаимной зависимостью. Отмечено, что задача назначения исполнителей и задача планирования должны решаться одновременно. Для этого была сформулирована многокритериальная задача, один из критериев которой (скорейшее завершение, формула 2.13) описывал задачу планирования, а остальные (обобщенные матричные критерии, формула 2.14) – задачу назначения исполнителей.

2. Анализ специфики предметной области и существующих решений позволил сделать вывод о том, что стандартные формулы оценки длительности отдельных задач (работ) не учитывают возможность их доработки и коррекции. В связи с этим, была предложена оценка длительности отдельных задач, учитывающая возможную коррекцию, возникающую как из-за несогласованности с заказчиком, так и в результате работы конкретного исполнителя, и обеспечивающая тем самым повышенную точность по сравнению с аналогами.

3. Учет специфики самого IT-проекта привел к выводу о необходимости реализации имитационной модели для учета вышеперечисленных нюансов и оценки вероятностно-временных характеристик случайной величины, описывающей длительность всего проекта. Данная модель позволяет на ранних этапах управления деятельностью по организации работ по конкретному проекту обеспечить повышенную точность по сравнению с аналогами.

Глава 3 АЛГОРИТМЫ РЕШЕНИЯ ЗАДАЧИ УПРАВЛЕНИЯ ПРОЕКТНОЙ ДЕЯТЕЛЬНОСТЬЮ IT-КОМПАНИИ

Глава посвящена алгоритмизации процесса управления деятельностью IT-компании путем организации работы ее исполнителей и планирования графика выполнения задач. Первая часть главы посвящена разработке обобщенного алгоритма решения данной задачи. Специфика назначения задач исполнителям приведена во второй части. Третья часть посвящена особенностям алгоритма коррекции работы компании. Исследование работы предложенных алгоритмов отражено в четвертой части главы. Пятая часть содержит основные выводы.

3.1 Обобщенный алгоритм управления IT-проектом

В главе 2 получена оценка длительности проекта, отличающаяся учетом коррекции задач, обеспечивающая повышенную точность по сравнению с аналогами.

Для уточнения перечня алгоритмов, которые необходимо разработать для решения поставленной многокритериальной задачи, сформируем диаграмму действий при создании IT-проекта (Рисунок. 3.1).

После поступления заявки от заказчика производится предварительная оценка IT-проекта с ориентировочными временными и стоимостными значениями. Если предварительная оценка проекта устраивает заказчика, наступает этап более детального планирования задач с учетом критерия минимизации критического пути. Одновременно осуществляется назначение для каждой из этих задач наилучшего с точки зрения обобщенного критерия исполнителя.

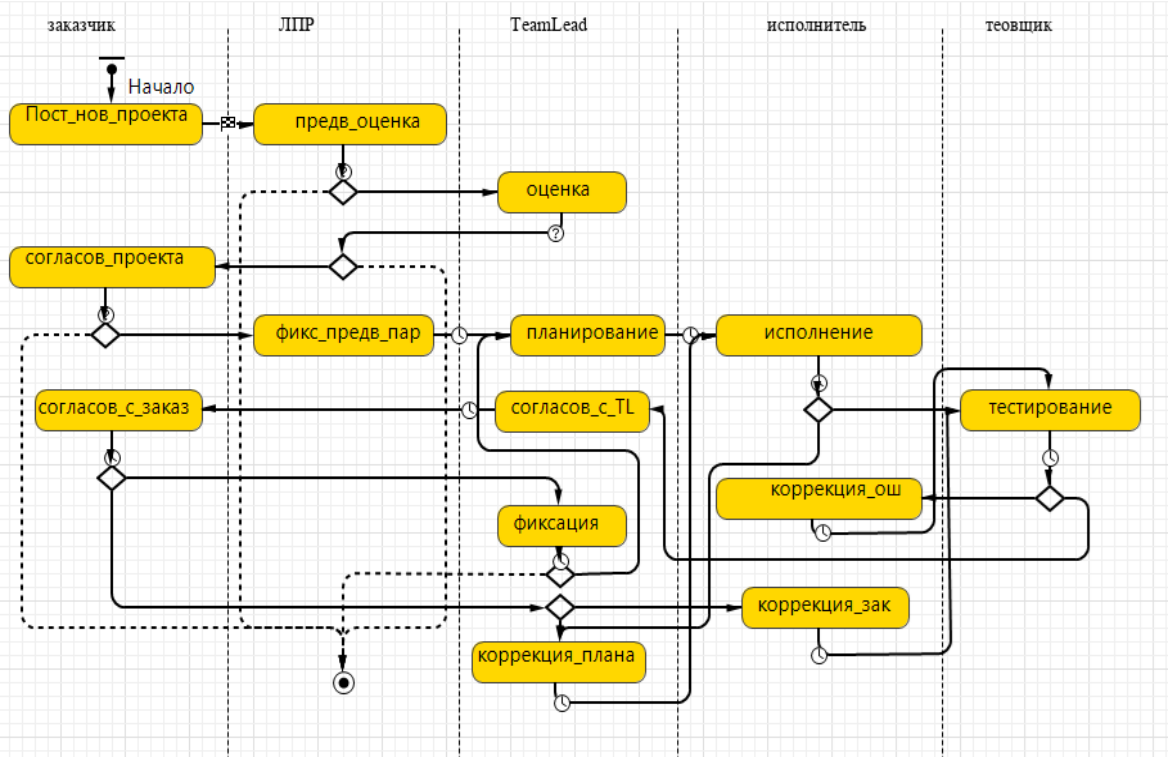


Рисунок 3.1 - Диаграмма действий при создании IT-проекта

Важным этапом при создании IT-проекта является учет возможной коррекции графика. В первую очередь, это может возникнуть в результате обсуждения решенных задач с заказчиком или изменения состава исполнителей. Также вполне вероятна коррекция задач в результате ее тестирования как изолированно, так и в комплексе с другими решенными задачами. Периодически осуществляется взаимодействие с заказчиком, демонстрация ему текущего положения IT-проекта (задач, выполненных к данному времени), в результате чего у него могут возникать разнообразные предложения. Если заказчика полностью устраивает некоторое множество решенных задач, они фиксируются как завершенные, и осуществляется переход к другим задачам. Проект завершается, когда последняя из его задач зафиксирована как решенная.

Представим алгоритмы для основных действий данной диаграммы.

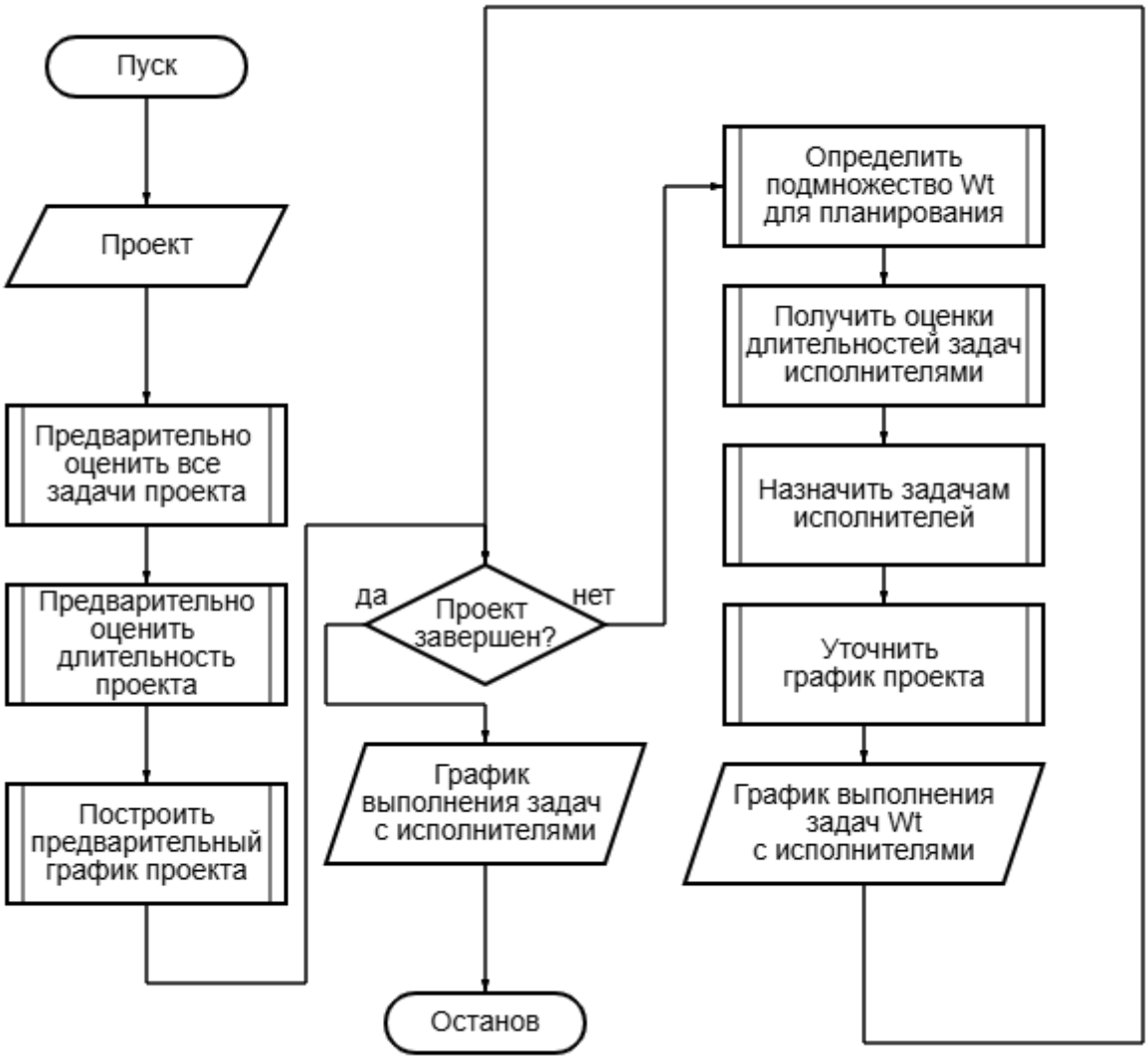


Рисунок. 3.2 - Обобщенный алгоритм

Поясним работу данного алгоритма.

Обобщенный алгоритм решения данной задачи заключается в построении предварительного графика проекта с точки зрения временного критерия

$$T_{\text{крит}} \rightarrow min, \tag{3.1}$$

а затем поэтапного его уточнения и закрепления исполнителей для каждой из задач.

Алгоритм позволяет точно подобрать исполнителей к задачам, учитывая разные критерии и их значимость. После каждого выбора он

пересчитывает приоритеты, чтобы добиться наилучшего распределения оставшихся ресурсов.

3.2 Алгоритм управления коллективом исполнителей

Рассмотрим более подробно процедуру определения соответствия между исполнителями и задачами. Подразумевается, что на вход алгоритма подается то подмножество задач, которое будет выполняться в данном временном интервале. Количество этих задач, скорее всего, превышает число исполнителей. Требуется сопоставить каждой задаче наилучшего исполнителя.

Как было отмечено ранее, основная сложность задачи заключается в том, что для выбора оптимального сопоставления требуется учесть несколько разных критериев. Часть этих критериев являются матричными, а другая часть – сложными нелинейными функциями. К таким критериям можно, например, отнести балансировку нагрузки исполнителей.

Рассмотрим особенности агрегации критериев. Существует несколько подходов к решению данной задачи. Будем использовать свертку критериев. Метод линейной свертки состоит из следующих этапов:

- приведение всех критериев к единой балльной шкале (так называемая нормализация критериев);
- определение весовых коэффициентов критериев;
- определение наилучшего значения.

Рассмотрим данные этапы более подробно. Поскольку разные критерии могут изменяться в разных диапазонах, требуется, чтобы вклад того или иного критерия в результат определялся лишь значением соответствующего весового коэффициента, а не принимаемыми значениями. Для этого все критерии нормализуются, т.е. приводятся к некоторой шкале. Диапазон этой шкалы должен быть одинаков для всех критериев. Вторая важная причина нормализации критериев заключается в разном стремлении отдельных

функций. Так, часть критериев может стремиться к максимуму, остальная часть – к минимуму. Если же использовать единый критерий, то требуется, чтобы все критерии были преобразованы по единому правилу (например, чем больше значение, тем лучше, или, наоборот, чем меньше, тем лучше).

Пусть единый интегральный критерий будет стремиться к минимуму. Тогда если некоторые критерии f_1 изначально стремился к минимуму, то его приведение к единому критерию будет выполняться по формуле:

$$\tilde{f}_l(y) = a + {(b-a)\cdot(y-y_1)}\Big/{(y_2-y_1)}. \quad (3.2)$$

В противном случае преобразование будет осуществляться следующим образом:

$$\tilde{f}_l(y) = b - {(b-a)\cdot(y-y_1)}\Big/{(y_2-y_1)}. \quad (3.3)$$

Здесь y_1 и y_2 – наименьшие и наибольшие возможные значения критерия $\tilde{f}_l$.

Для приведения критериев к единой балльной шкале используется формула:

$$F(x) = \sum_{i=1}^{L} \alpha_i \cdot \tilde{f}_i(x). \quad (3.4)$$

Здесь α_i – коэффициенты, показывающие значимость каждого критерия. Она задаются лицом, принимающим решения; $\tilde{f}_i$ – нормализованные критерии.

Рассмотрим подход к решению поставленной задачи с точки зрения единого критерия. Как было отмечено в главе 1, в случае матричных критериев (или сведении всех критериев к матричному) можно воспользоваться Венгерским методом и его разновидностями. Однако, возникает сложность в случае, если критерий не является матричным и если он не может быть сводим к решению подзадач (т.е. изначально с точки зрения данного критерия задача не обладает свойством оптимальной подструктуры). К таким критериям относится, например, балансировка загрузки исполнителей, которая описана с помощью формул (2.4) и (2.5).

Без ограничения общности будем рассматривать случай одинаковой загрузки, хотя, возможны разные вариации (например, когда загрузка по разным классам исполнителей разная).

Сложность данного критерия заключается в том, что изначально (до начала планирования) загрузка будет нулевая. По мере назначения задач исполнителям загрузка будет меняться, но окончательный результат будет ясен лишь в случае, когда исполнители будут назначены всем задачам.

В связи с этим, для решения поставленной многокритериальной задачи будем использовать комбинацию из двух подходов. Изначально с помощью эвристик попытаемся найти решение задачи закрепление за исполнителями каждой из задач, а затем улучшить найденное решение с помощью вариации методов локального поиска.

Рассмотрим каждый из этапов более подробно. На первом этапе необходимо выполнить нормализацию. Пусть, без ограничения общности, будем использовать шкалу [0,1]. Тогда для нормализации необходимо:

- оценить возможные наибольшие и наименьшие значения каждого из критериев;

- перевести текущие значения с помощью формулы (3.2) или (3.3).

После нормализации необходимо для каждого критерия подобрать веса. В классической интерпретации методов агрегации критериев (например, метода линейной свертки), веса должны удовлетворять следующим ограничениям:

$$\begin{cases} \alpha_i \geq 0; \\ \sum_{i=1}^{L} \alpha_i = 1 \end{cases} \quad (3.5)$$

С учетом того, что исследуемая задача отличается от классической, будем веса назначать динамически. Это означает, что на каждом шаге в зависимости от текущей характеристики задачи и исполнителя вес будет меняться.

Каждый из таких коэффициентов изначально можно подобрать в зависимости от специфики критерия. Например, для критерия скорейшего

завершения работ (пусть без ограничения общности, это будет критерий под номером 1) вес целесообразно рассчитать по формуле:

$$\tilde{\alpha}_{ij}^{(1)} = k_1 \cdot {}^{1}/_{(1+r_i)}, \quad (3.6)$$

Где k2 – глобальный статический вес данного критерия; r_i – временной резерв работы, рассчитанный по формулам метода критического пути. В случае такого расчета веса для работ, стоящих на критическом пути, когда $r_i=0$, коэффициент становится равным 1, и важность данного критерия доминирует.

Для балансировки нагрузки (пусть это будет критерий 2) необходим коэффициент, который был бы достаточно велик, если исполнитель недогружен (стимулируя, тем самым, распределять задачи между исполнителями, которые имеют в настоящий момент недостаточную нагрузку). Выберем для этого экспоненциальный критерий, который определяется формулой:

$$\tilde{\alpha}_{ij}^{(2)} = k2 \cdot e^{-\delta\left(L_j - \bar{L}\right)}. \quad (3.7)$$

Здесь k2 – глобальный статический вес данного критерия балансировки; δ - параметр, который определяет, насколько сильно должен увеличиваться или уменьшаться критерий при приближении к среднему значению и отклонении от него. На рисунке 3.3 приведен пример графиков изменения данного весового коэффициента для значений δ=1 (ряд 1) и δ=0.5 (ряд 2) со средним значением нагрузки, равным 4 условных единицы. Как видно из этого рисунка, чем больше задать значение δ, тем больше будет значение данного коэффициента для исполнителей, которым еще не назначили задачи или назначили их в недостаточном количестве.

Также можно настраивать другие критерии в соответствии с их смысловой нагрузкой. В целом возможно, чтобы данные критерии не удовлетворяли условию (3.5).

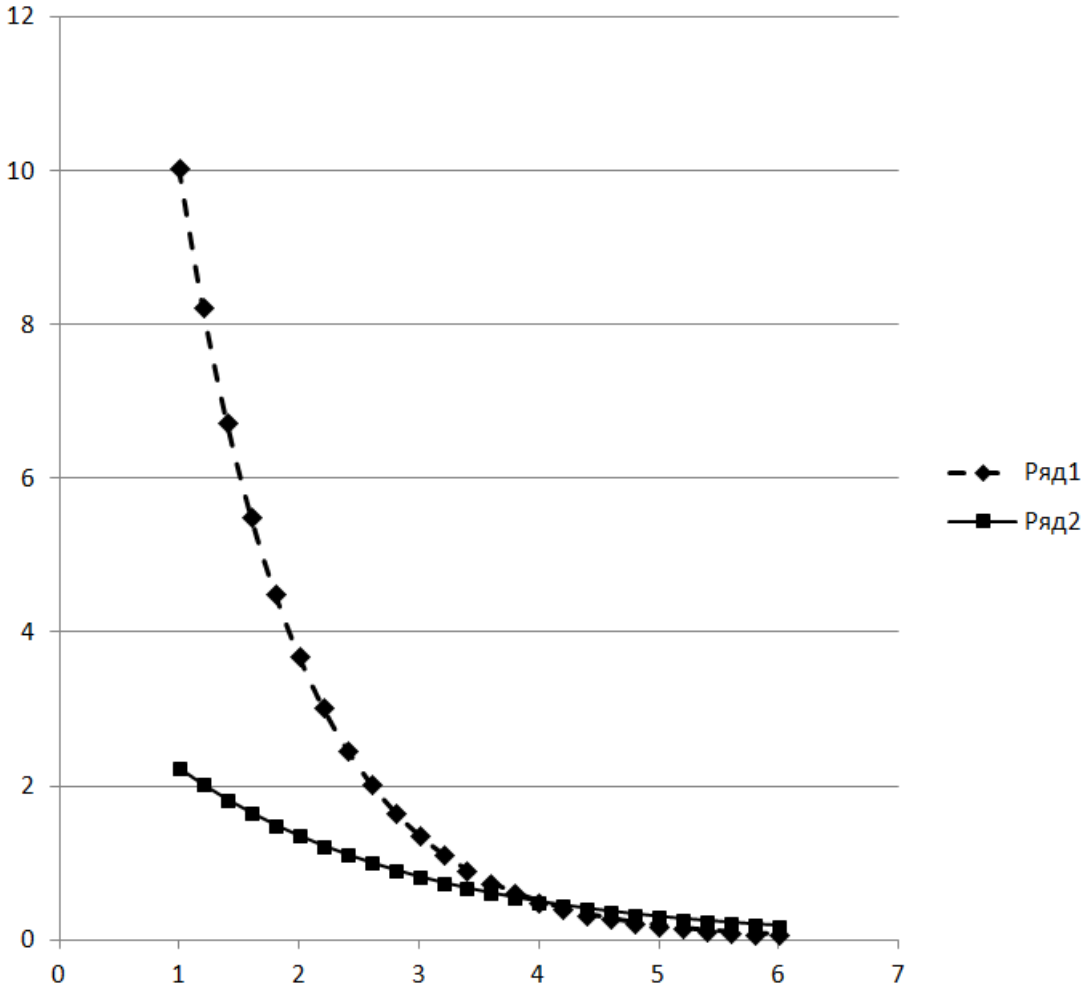


Рисунок 3.3 - График изменения коэффициента балансировки нагрузки для разных значений δ

Однако, если добиваться этого, то необходимо их нормализовать:

$$\alpha_{ij}^{(k)} = \tilde{\alpha}_{ij}^{(k)} \Big/ \sum_{l=1}^{L} \tilde{\alpha}_{ij}^{(l)}, \qquad (3.8)$$

где L – число критериев.

Как можно видеть из формул (3.6) – (3.8), предполагается, что веса будут представлять собой матрицу, каждый элемент которой будет определять значение, соответствующее конкретной паре – «задача» - «исполнитель». Данные веса следует динамически переопределять на каждом шаге.

Далее для каждого исполнителя и каждой из рассматриваемых на данном этапе задач определим значение целевой функции по формуле:

$$F_{ij} = \sum_{l=1}^{L} \tilde{\alpha}_{ij}^{(l)} \cdot \tilde{f}_{ij}^{(l)}. \quad (3.9)$$

Здесь $\tilde{\alpha}_{ij}^{(l)}$ - значение весовых коэффициентов критерив, рассчитанных по формуле (3.8), $\tilde{f}_{ij}^{(l)}$ - нормализованные значения критерия *l*, рассчитанные для данной задачи i и данного исполнителя j.

Основным этапом первой части алгоритма является применение жадной эвристики к парам (исполнитель, задача) с целью выявления наилучшей (на данном шаге) комбинации. Данная эвристика заключается в выборе на каждом этапе такого соотношения, для которой выбор любого другого решения в наибольшей степени ухудшит итоговый результат. Таким образом, каждый раз выбирается пара, которая является наилучшей с точки зрения сравнения альтернатив. Для реализации данной эвристики на основе полученной матрицы *F*, описывающей интегральный критерий, предлагается следующее преобразование:

$$f_{ij}^{\text{нов}} = \min_{\substack{1 \le k \le n \\ k \ne i}} F_{kj} + \min_{\substack{1 \le k \le n \\ k \ne j}} F_{ik} - 2F_{ij}. \quad (3.10)$$

Опишем смысл предлагаемой формулы. Разница между минимальным значением в строке (или в столбце) и текущим значением показывает, насколько возможно было бы улучшить/ухудшить результат, если бы не выбирать соответствие между задачей j и исполнителем i. Если результат отрицательный, то выбрав другие соответствия, критерий ухудшится. Если положительный, то, наоборот, видим, насколько предпочтительнее данный выбор, чем выбор других соответствий между исполнителем и задачей.

Выбрав с помощью формулы (3.10) наилучшее соответствие, убираем из рассмотрения соответствующую задачу и исполнителя, и продолжаем так до тех пор, пока задачи или исполнители на данном временном интервале не закончатся.

Если остались некоторые незапланированные задачи, они переносятся на следующий период.

Важным нюансом данного метода является пересчет критериальных значений и весов на следующем временном интервале. Например, если некоторую работу перенесли на более поздний срок, у нее уменьшился резерв. Следовательно, с точки зрения данного критерия она должна быть более приоритетной. Также, если на данном этапе у некоторого исполнителя существенно увеличилась нагрузка (например, назначена очень большая по времени задача), то с помощью критерия (3.9) можно уменьшить вероятность выбора именно данного исполнителя и, следовательно, увеличить возможность закрепления задач за другими менее занятыми специалистами.

Недостатком данного подхода является отсутствие гарантии оптимального решения. Для увеличения вероятности этого события применяется метод имитации отжига. Его суть в данном алгоритме сводится к следующему:

- выбирается некоторый временной интервал;

- выбираются две произвольные задачи;

- производится обмен исполнителями, закрепленными за этоми задачами;

- пересчитываются параметры и критериальная функция;

- далее производится сравнение значение критерия на новом решении с эталонным вариантом: если оно лучше, то оно становится эталонным, в противном случае это производится с определенной вероятностью.

Указанные выше действия необходимо повторять в цикле. Параметром цикла является аргумент, имитирующий температуру. Цикл повторяется от максимальной до минимальной температуры. Существует несколько способов изменения температуры. Выберем простейший способ уменьшения температуры на некоторый шаг h.

Вероятность принятия нового решения в случае, если оно оказалось хуже эталонного рассчитывается по формуле:

$$P = \exp(-\Delta F/t). \quad (3.11)$$

В формуле (3.11) ΔF – это разница между новым и эталонным значением функции; t – текущая температура.

Описанные выше шаги выполняются циклически с целью улучшить найденные решения. Если очередная смена привела к улучшению целевой функции, она берется за эталон. После заданного числа шагов цикла выводится наилучший результат. Это и будет являться выходом алгоритма, позволяющего закрепить за каждой задачей наилучшего с точки зрения множества критериев исполнителя.

Схема данного алгоритма приведена на Рисунке 3.4. Как видно из данного рисунка, на вход алгоритма поступает множество задач, которые надо сделать в данном временном интервале (Tasks()), и множество исполнителей (Exec). Первый крупный этап алгоритма заключается в эвристическом поиске соответствий согласно описанному выше подходу. Результаты на каждом шаге эвристического алгоритма заносятся в массив bestpair, каждый элемент i которого содержит идентификатор исполнителя, который будет выполнять задачу i. Второй этап (правая часть алгоритма) заключается в попытке улучшения найденного решения методом имитации отжига. Выбрав временной интервал ΔT и две произвольные работы на нем i1 и i2, алгоритм меняет исполнителей этих работ, занося новые значения в массив chpair. Далее производится сравнение значения функции на старом и новом наборе и, при необходимости, переход к новому решению. В операторе сравнения константа r – это случайно сгенерированное число в диапазоне (0,1).

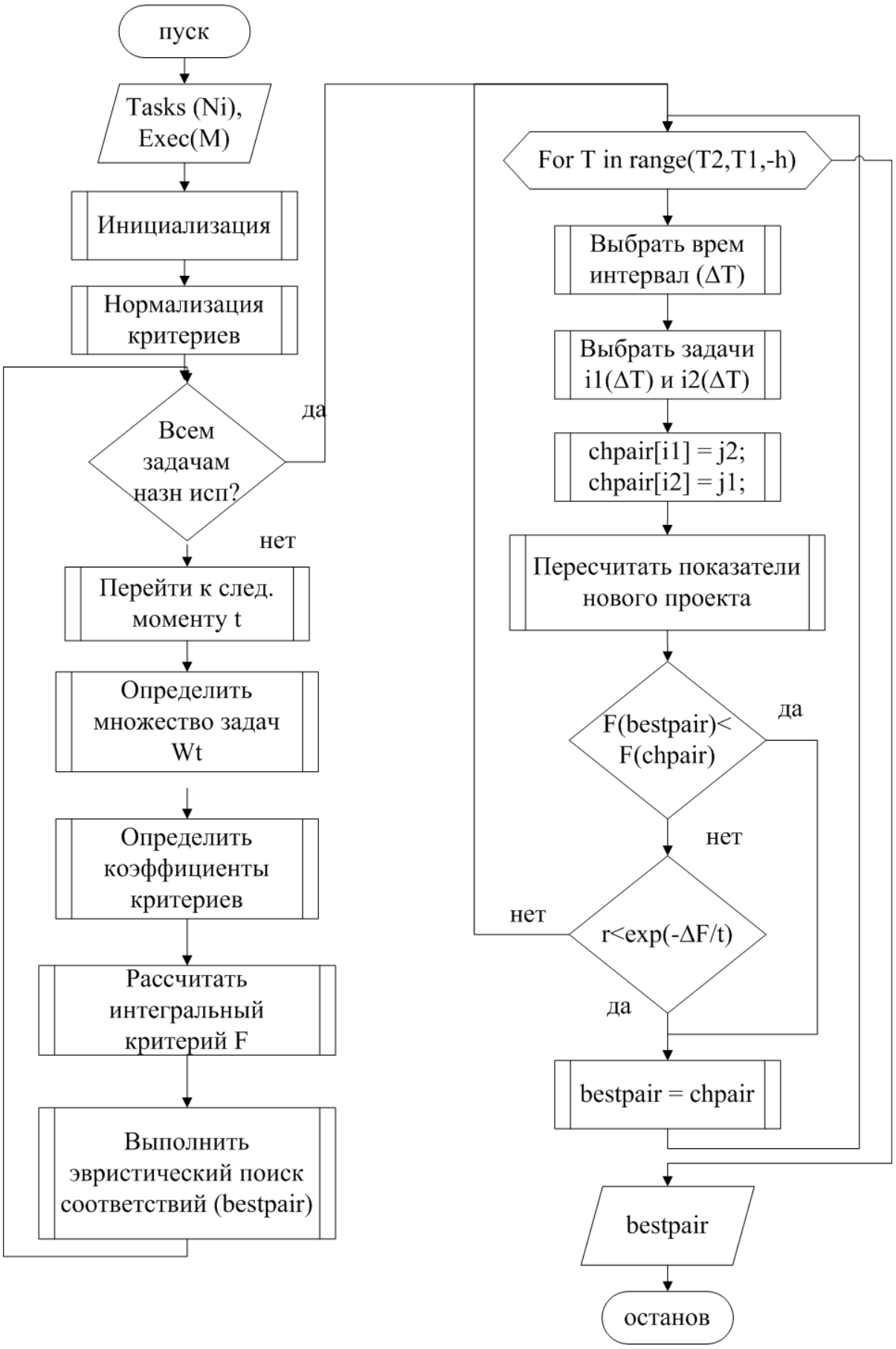


Рисунок 3.4 – Алгоритм назначения исполнителей

3.3 Алгоритм коррекции существующего план-графика

Одной из ключевых особенностей исследуемой задачи является ее динамичность. Изменения могут происходить с IT-проектом на любых стадиях его исполнения. Проанализируем диаграмму, представленную на Рисунке 3.1.

Как можно видеть, результат выполнения проекта на всех его стадиях предъявляется заказчику для одобрения им выполненных задач и согласования новых. В результате решаемые задачи могут существенно корректироваться, что вносит серьезные изменения в оценки длительностей их решения, а также появляться новые задачи. Заказчик может потребовать быстрого решения некоторых задач, изменив, тем самым приоритеты выполнения задач. Кроме того, время выполнения каждой задачи разными исполнителями существенно отличается из-за разной квалификации исполнителей IT-команды. Также следует отметить периодическое возникновение так называемых нестандартных задач, с которыми данная команда исполнителей ранее не сталкивалась. В связи с этим, предсказать заранее длительность решения каждой задачи проекта крайне сложно. Кроме того, исполнитель может заболеть, уехать в экстренную командировку, что потребует перепланировать назначенные ему задачи другому специалисту. Все эти факторы не позволяют использовать классические методы для организации графика по выполнению работ IT-проекта.

Выделим основные события, которые потребуют оперативного реагирования и динамического перепланирования в IT-проекте:

- временная недоступность исполнителя (болезнь/командировка);
- уход/появление новых исполнителей;
- появление новых задач в результате очередного обсуждения с заказчиком/тестирования им работы текущих выполненных задач;
- корректировка представленных заказчику задач;
- корректировка длительностей задач;

- перераспределение какой-либо задачи другому исполнителю.

Исходя из данных возможных событий, выделим два направления коррекции:

- текущие задачи должны быть перераспределены (болезнь исполнителя, исполнитель по той или иной причине не может справиться с задачей);

- решаемые задачи не подвергаются коррекции, изменения касаются лишь тех задач, которые еще не начаты.

Основная идея алгоритма коррекции заключается в непрерывном мониторинге состояния проекта, оперативного реагирования на внештатные события и сбор статистики и, возможно, выдача рекомендаций ЛПР для принятия определенных решений (например, для принятия нового исполнителя и т.д.).

Общая схема алгоритма коррекции представлена на Рисунке 3.5. Рассмотрим данный алгоритм более подробно. На вход данного алгоритма поступает множество задач, завершенных или выполняемых к настоящему моменту времени T. Каждая задача имеет свой статус, который может принимать одно из следующих значений:

0 – задача не принята к исполнению;

1 – задача запланирована и исполняется;

2 – задача готова для согласования с заказчиком;

3 – задача находится на стадии коррекции;

4 – задача завершена.

Будем считать задачу завершенной, если заказчик в результате согласования удовлетворен соответствующим функционалом.

Первый шаг алгоритма заключается в фиксации всех начатых задач. Без ограничения общности, это задачи множества W_1. Далее, в зависимости от события, вызвавшего коррекцию, обновляются ограничения задачи. Это может быть измененное число исполнителей, измененное время задачи и т.д.

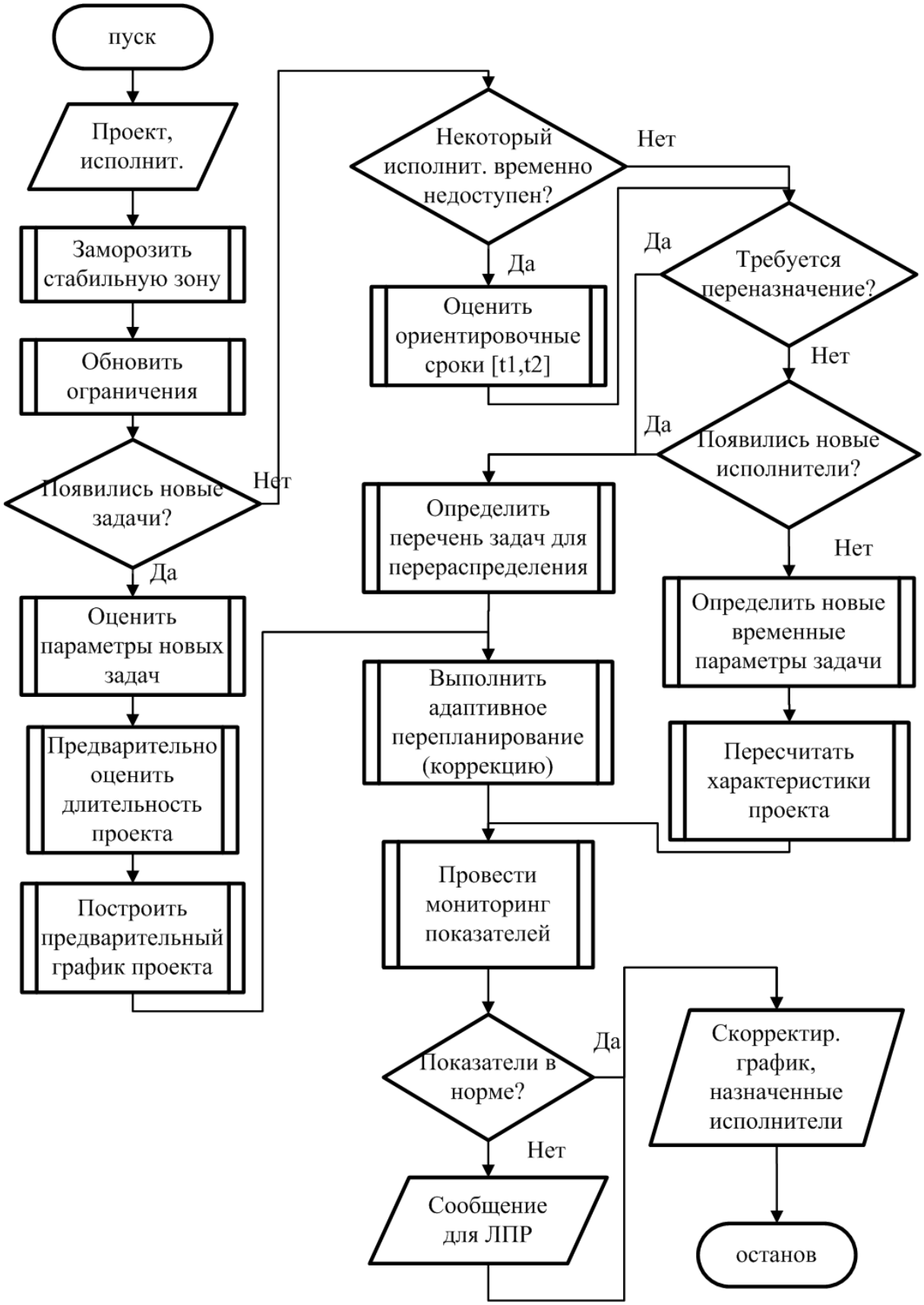


Рисунок 3.5 – Общий алгоритм коррекции

После этого последовательно проверяются все события, которые могли привести к коррекции и осуществляется реакция на конкретное событие. Если появились новые задачи, то определяются их параметры, оценивается новая длительность, после чего.

Рассмотрим момент T очередного завершения согласования задач. Будем считать, что планирование производится на некоторый период Δt. Предметом интереса являются все незавершенные задачи. Разделим задачи на подмножества, согласно их статусу. Подмножество W_0 будет содержать задачи, которые еще не начаты, но могут быть начаты в данный момент. Кроме того, в это множество будут входить все задачи, которые могут быть начаты позднее, но не позже момента $T+\Delta t$ (из-за завершения исполняемых в данное время задач). Подмножество W_2 будут составлять задачи, которые готовы для представления их заказчику. Подмножество W_3 будет состоять из задач, которые необходимо корректировать, а подмножество W_1 – это те задачи, которые выполняются в данный момент. Будем считать задачу завершенной, если заказчик в результате согласования удовлетворен соответствующим функционалом. В связи с этим, в данный момент времени будут интересны задачи $W_0 \cup W_1 \cup W_2 \cup W_3$. В первую очередь, с закзачиком согласуются все задачи из W_2. В результате согласования некоторой задачи возможны 2 ситуации:

- заказчик просит скорректировать задачу;
- заказчика полностью устраивает представленная задача.

Следовательно, задачи из множества W_2 будут подразделены на два подмножества: W_{21} (готовых задач), которое не будет интересно для планирования, и W_{22} (задач, которые необходимо в той или иной степени исправить).

Кроме того, в результате согласования может появиться новое множество $W_н$ задач, которые не были раньше согласованы, но которые заказчику потребовались на данном этапе.

Длительности задач из последних двух подмножеств, если они не пусты, необходимо оценить. Следующий важный момент в современных IT-проектах заключается в том, что заказчик, как правило, сам определяет, какие задачи необходимо на данном этапе выполнить в первую очередь (даже если ранее было оговорено другое). Таким образом, его мнение для изменения приоритета решаемых задач является ключевым. Если же заказчику не важно, в каком порядке будут выполняться работы, распределение последовательности решения задач будет разниматься лицо принимающее решения (ЛПР) - team lead, руководитель проекта и т.д. Согласно выставленному приоритету необходимо из тех задач, к выполнению которых можно приступить в данный момент планирования T и которых, как правило, гораздо больше, чем исполнителей, выбрать перечень задач для исполнения в данный момент. Остальные задачи будут перенесены на более поздний период.

Следующим важным этапом является распределение заданий по исполнителям. Как было отмечено ранее, одной из ключевых особенностей является разный уровень исполнителей. Задачи, как правило, также отличаются разным уровнем сложности. В связи с этим, целесообразно распределить задания таким образом, чтобы достичь некоторый суммарный оптимум качества. Это фактически обобщение задачи о назначениях. Для ее решения можно определить одну или несколько целевых функций, наиболее важных для данного проекта, затем каким-либо образом выполнить свертку критериев, перейдя к однокритериальному случаю. Следует отметить, что в данном случае нет ограничения на количество задач, выполняемых исполнителем в планируемый период. Более того, в ряде случаев целесообразно запланировать исполнителю несколько задач, чтобы у него к следующему моменту планирования была бы возможность осветить нюансы задач, обсудить их с заказчиком и заложить основу для их дальнейшего решения.

На этом основной процесс планирования на данном этапе завершается. Предполагается, что данные этапы будут осуществляться через определенные моменты времени Δt. Это позволит, с одной стороны, выделить действительно важные к настоящему моменту задачи для исполнения, с другой – на каждом этапе учесть актуальные моменты. Кроме того, такой способ планирования учитывает актуальный график работы сотрудников (например, их отпуска, болезни и т.д.).

Существенным минусом данного подхода к планированию является невозможность заранее точно сказать планируемый срок завершения проекта. Однако, для получения проекта все равно необходимо оценивать эти характеристики. Следовательно, вначале заказчику предоставляются ориентировочные сроки выполнения проекта. Но появление новых задач, а также существенная коррекция существующих может привести к существенному увеличению временных показателей. В связи с этим, при планировании следует также оценивать степень отклонения новых сроков проекта от первоначальных значений. Если изменения в проекте на данном этапе значительны, целесообразно заново оценить длительность проекта для уведомления заказчика.

Таким образом, разработана общая схема коррекции план-графика выполнения работ на некотором временном интервале T. Она представлена на Рисунке 3.6.

Рассмотрим метрики, которые необходимы для анализа критичности внесенных изменений. К ним можно отнести следующие:

- перегруз исполнителей;
- количество смен исполнителя по задачам;
- относительная стабильность проекта (заключающаяся в сравнительно небольшом отклонении планируемого и фактического времени);
- выполнение вех/дедлайнов.

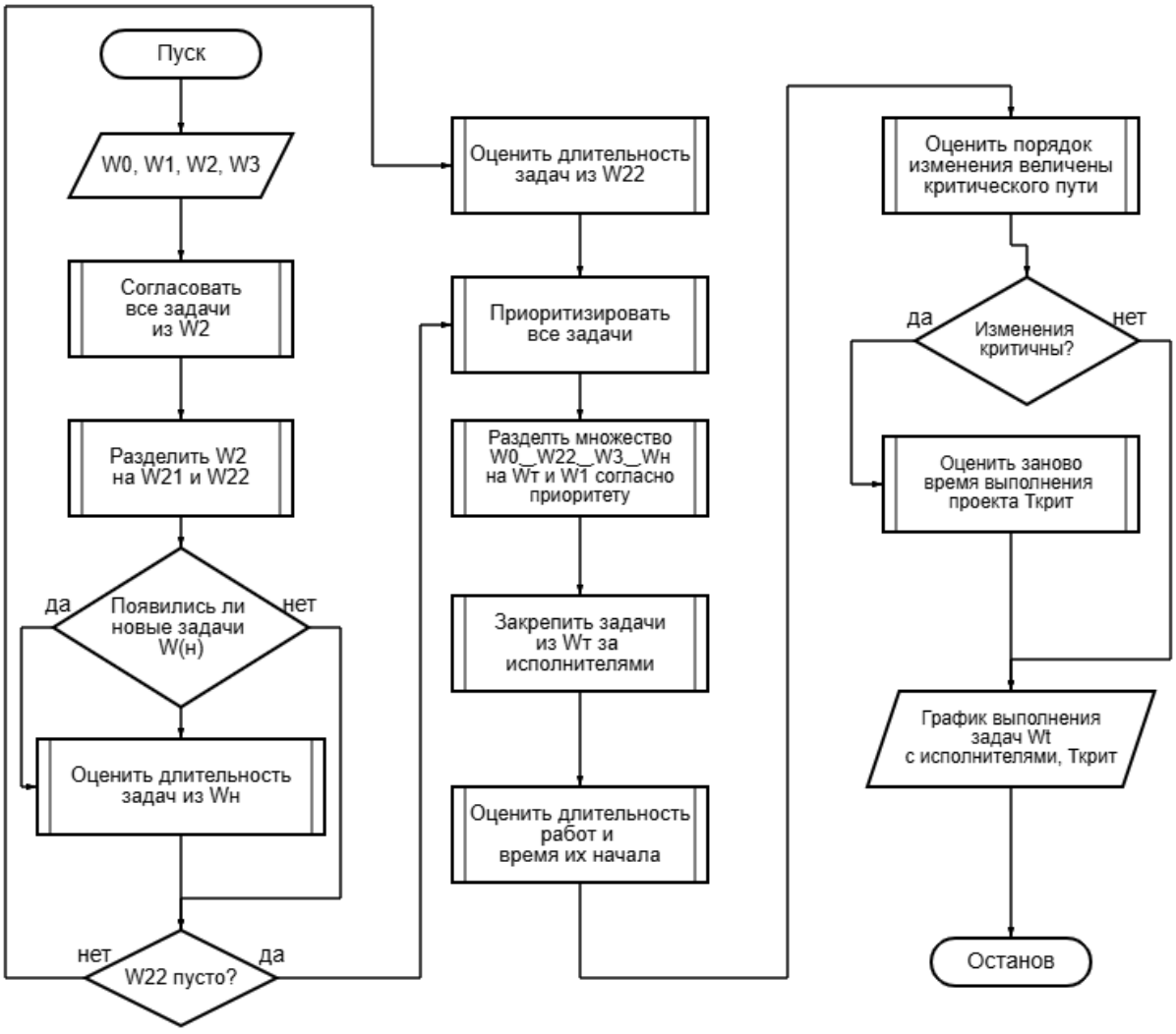


Рисунок 3.6 - Схема алгоритма коррекции план-графика в некоторый момент T

Рассмотрим эти задачи более подробно. Перегруз исполнителей может негативно влиять на их деятельность, поскольку приводит к выгоранию специалистов. Данная величина определяется по каждому сотруднику как доля рабочего времени, занятого задачами. Для ее расчета можно использовать формулу:

$$Load(j,T) = \sum_t \sum_i x_{ij} \cdot dur_{ij} \Big/ T_{\text{раб}}. \quad (3.12)$$

Здесь T – исследуемый период; нагрузка суммируется по всем работам, выполняемым в течение этого периода; $T_{\text{раб}}$ – рабочее время данного периода (для исключения выходных и праздничных дней). Данное значение по

каждому исполнителю необходимо сравнить с показателем, приемлемым данным ЛПР (например, 85 или 90%). В случае, если по какому-либо исполнителю возникает перегруз, это сигнал о том, что необходимо снизить его нагрузку. Если перегруз возникает у всех исполнителей,

Количество смен исполнителей по задачам показывает, как часто задаче необходимо переназначить исполнителя. Как было отмечено в алгоритме, это действие выполняется в двух случаях:

- когда исполнитель временно нетрудоспособен;
- когда исполнитель не может справиться с задачей.

Очевидно, что при переназначении задачи новый исполнитель должен потратить определенное время, чтобы изучить задачу. Поэтому если какой-либо сотрудник перегружен такими изменениями, его продуктивность будет в целом снижена по сравнению с работой без прерываний над задачами с самого начала. Поэтому данную метрику целесообразно рассчитывать сначала по отдельному исполнителю по формуле:

$$ST = {}^{cnt\_sw}/_{tot\_cnt}. \quad (3.13)$$

Здесь cnt_sw - число задач за исследуемый период, которые были перенаправлены исследуемому исполнителю в прерванном виде (в результате переназначения); tot_cnt - общее число задач, закрепленных за данным исполнителем за данный период.

Если данный показатель у некоторого исполнителя нарушает предельные значения, требуется увеличить штраф в целевой функции за переназначение. Для ЛПР такие показатели являются сигналом к тому, чтобы учесть данную особенность при последующем назначении (например, некоторый исполнитель пока не готов решать подобные задачи).

Относительная стабильность проекта будет измеряться как разница между фактическим и плановым временем проекта. Для измерения данной метрики для каждой задачи необходимо рассчитать разницу между

фактическим и планируемым временем ее окончания и далее взять 95-й перцентиль (т.е. почти максимальное смещение).

Например, если брать значения отклонений [1,2,3,10,12], то вычислив k по формуле:

$$k = 0.95 \cdot (5 - 1) + 1 = 4.8;$$

$$P95 = \Delta_{(4)} + (k - \lfloor k \rfloor) \cdot \left(\Delta_{(5)} - \Delta_{(4)}\right) = 10 + (5 - 4.8) \cdot (12 - 10)$$

Если данная метрика превышает порог, то ЛПР должен решать стратегические вопросы по возникновениям причин такого превышения и принимать соответствующие решения (например, при частых изменениях со стороны заказчика согласовать или более редкий ритм изменений или вообще провести «заморозку требований»; если причина в оценках длительностей задач, необходимо внести новые составляющие в формулы и скорректировать оценки; в ряде случаев перераспределить задачи или переназначить некоторых исполнителей вручную).

Выполнение вех/дедлайнов является главным показателем для заказчика. Поэтому контрольные вехи желательно выполнять в срок. Для некоторой вехи j выполнение ее в срок оценивается по формуле:

$$MonT_j = max(0; date\_fact(j) - dedline(j)). \quad (3.14)$$

Если вероятность срыва вехи превышает некоторое пороговое для ЛПР значение, то можно алгоритмически или вручную перераспределить исполнителей, привлечь какие-либо временные ресурсы и т.д. В любом случае необходим анализ причины срыва с целью недопущения этого в будущем.

Таким образом, в целом необходимо исследовать метрики и принимать решения, представленные в Таблице 3.1

Таблица 3.1 - Исследуемые метрики и принимаемые решения

Метрика	Что означает	Действие алгоритма	Действие teamlead и ЛПР
Перегруз исполнителей	Насколько сбалансировано распределена нагрузка; риск выгорания; достаточно ли исполнителей для данного проекта	Увеличить коэфф-т критерия балансировки нагрузки (для необходимых исполнителей) и перезапустить алгоритм в исследуемом временном интервале	При постоянной перегрузке привлечь новых исполнителей
95 перцентиль	Относительная стабильность проекта (для IT-компании)	Снизить частоту пересчетов; увеличить коэффициент критерия скорейшего завершения для задач, являющихся источником нестабильности и перезапустить алгоритм	согласовать или более редкий ритм изменений или «заморозить требования»; скорректировать оценки длительностей; перераспределить задачи или переназначить некоторых исполнителей вручную
Количество смен исполнителя по задачам	Устойчивость работы коллектива	Внести изменения в показатели исполнителя, не справляющегося с задачами	Анализ причин смен; переназначить некоторых исполнителей вручную
Выполнение в срок вех/дедлайнов	Надежность IT-организации с точки зрения заказчика	Внести изменения в коэффициент критерия завершения проекта в срок и перезапустить алгоритм в исследуемом временном интервале	Поиск причины невыполнения в срок; перераспределение вручную; привлечение дополнительных ресурсов

3.4 Исследование работы алгоритмов управления деятельностью IT-компании

Рассмотрим результаты работы описанных выше алгоритмов. Без ограничения общности, будем рассматривать планирование по уже свернутому единому критерию, который требуется минимизировать. Рассмотрим специфику работы на графе, приведенном на рис. 3.7. В рассмотренном примере в качестве такого единого критерия используется время.

3.4.1. Анализ работы алгоритма без коррекции

Проанализируем сначала эффективность предложенных алгоритмов управления организационной системой без учета коррекции отдельных задач. Для этого проведем серию экспериментов, каждый из которых заключается:

- в решении задачи распределения работ за исполнителями и оценке времени их начала в предположении, что каждая задача не потребует последующей коррекции;
- произвольном назначении исполнителей;
- сравнительном анализе получившихся результатов.

Рассмотрим один из экспериментов на следующем графике, описывающем последовательность выполнения задач (Рисунок. 3.7).

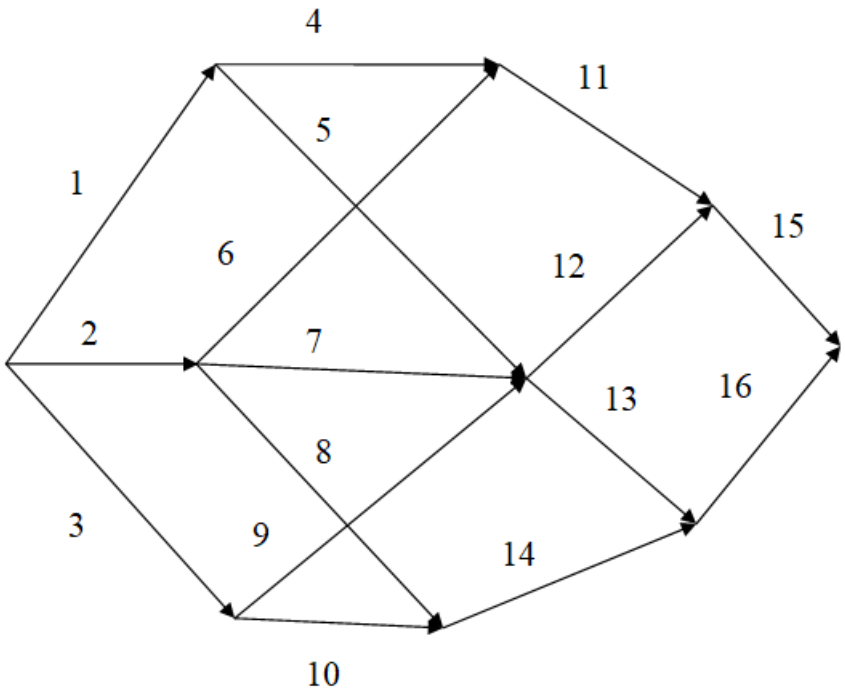


Рисунок 3.7 – Пример сетевого графика

На весах графа показаны идентификаторы задач. Пусть, без ограничения общности, имеются 4 исполнителя для выполнения данных задач.

По формуле 2.9 рассчитаем ориентировочную длительность выполнения каждым исполнителем каждой из задач. Результате представлены в Таблица. 3.2.

Таблица 3.2 - Оценки длительностей выполнения каждой из задач каждым исполнителем

Задача	Исполнит. 1	Исполнит. 2	Исполнит. 3	Исполнит. 4
1	6,667	8,000	7,667	5,833
2	8,667	6,833	7,667	7,000
3	9,333	7,167	10,000	9,000
4	7,000	9,167	7,833	7,000
5	8,000	8,833	9,000	7,833
6	7,167	9,000	8,000	7,167
7	9,833	7,833	9,000	8,000
8	10,000	8,000	9,167	7,833
9	5,167	6,000	8,000	4,833
10	5,833	6,833	6,833	5,167
11	7,833	6,833	7,167	6,000
12	8,000	7,000	7,000	7,000
13	9,000	7,833	8,167	8,667
14	7,167	9,000	8,167	7,667
15	6,167	8,000	7,833	6,000
16	7,833	9,000	9,833	7,000

На предварительном этапе планирования определяется раннее, позднее время начала задач и их резерв. При этом следует отметить, что, с учетом того, что пока назначение исполнителей не было выполнено, оценки длительностей пока неизвестны. В связи с этим, воспользуемся формулой 2.11, если известно некоторое вероятностное распределение исполнителей по задачам или формулой 2.12, если такой информации нет. Предположим, что проект достаточно новый, и какая-либо информация о вероятности назначения той или иной задачи каждому из специалистов ответствует. Тогда для временных оценок воспользуемся формулой среднего значения, в результате чего получим следующие предварительные характеристики (Таблица 3.3).

Таблица 3.3 - Предварительные временные характеристики для исследуемого сетевого графика

Задача	средняя длит	T_i раннее	T_i позднее	резерв
1	7,042	0	0,748	0,748
2	7,542	0	0	0
3	8,875	0	1,333	1,333
4	7,750	7,042	11,334	4,292
5	8,417	7,042	7,791	0,748
6	7,834	7,542	11,250	3,708
7	8,667	7,542	7,542	0
8	8,750	7,542	7,875	0,332
9	6,000	8,875	10,208	1,333
10	6,167	8,875	10,458	1,583
11	6,958	15,375	19,084	3,709
12	7,250	16,208	18,792	2,584
13	8,417	16,208	16,208	0
14	8,000	16,292	16,625	0,332
15	7,000	23,458	26,042	2,584
16	8,417	24,625	24,625	0

В данной таблице во втором столбце приведена ориентировочная длительность выполнения каждой задачи, в третьем – раннее время решения задачи, в четвертом – позднее время решения. Последний столбец показывает временной резерв задачи, который потребуется в случае, когда необходимо делать отбор тех задач, которые надо выполнить в данный момент, и тех, которые следует перенести на более поздний период. Данная информация является весьма грубой, и необходима лишь для того, чтобы ориентироваться в первоначальных временных резервах, которые в дальнейшем будут уточняться, во-первых, при планировании работ, во-вторых, при уточнении времени их завершения и последующей коррекции.

Далее будем итеративно проходить по всем временным интервалам и назначать исполнителей, согласно алгоритму, представленному на Рисунке 3.3.

На первом шаге можно выполнить задачи 1,2 и 3, т.к. все остальные работы будут последующими. Приведем временные характеристики данных задач для каждого из исполнителей (Таблица 3.4).

Таблица 3.4 - Временные значения для первых трех работ

Задача	Исполнит. 1	Исполнит. 2	Исполнит. 3	Исполнит. 4
1	6,667	8,000	7,667	5,833
2	8,667	6,833	7,667	7,000
3	9,333	7,167	10,000	9,000

Построим ориентировочную матрицу выигрыша для каждой задачи j в случае, если бы ей не был назначен исполнитель i. Для этого найдем, как указано в алгоритме для каждого значения максимум из оставшихся элементов в строке (временные затраты другого исполнителя – наилучшего для данной задачи) и в столбце (временные затраты данного исполнителя при выполнении другой задачи) и вычтем двойное значение временных затрат данного исполнителя для данной задачи. В частности, для матрицы, приведенной в Таблице 3.4, такая матрица выигрыша будет иметь вид, приведенный в Таблице 3.5.

Таблица 3.5 - Матрица выигрыша

Задача	Исполнит. 1	Исполнит. 2	Исполнит. 3	Исполнит. 4
1	1,166	-3,334	-1,834	2,001
2	-3,834	0,501	-0,834	-1,334
3	-4,832	1,499	-5,166	-5

Найдем наибольшее значение в данной матрице и соответствующие этому значению строку и столбец. Назначим исполнителю 4 задачу 1, вычтем из матрицы первую строку и четвертый столбец.

Таблица 3.6 - Матрица после назначения исполнителя 4

Задача	Исполнит. 1	Исполнит. 2	Исполнит. 3
2	8,667	6,833	7,667
3	9,333	7,167	10,000

Сформировав для данной таблицы аналогичную матрицу, получим.

Таблица 3.7 - Матрица выигрыша для трех исполнителей

Задача	Исполнит. 1	Исполнит. 2	Исполнит. 3
2	-1,168	1,168	1,499
3	-2,832	1,832	-5,166

Назначим исполнителю 2 задачу 3. Оставшуюся задачу делает исполнитель, у которого значение целевой функции лучше, в этом случае это исполнитель 3. Таким образом, после первого шага получим следующее закрепление (Таблица 3.8).

Таблица 3.8 - Закрепление задач за специалистами после первого шага алгоритма

Специалист	Задача	Вр нач	Вр оконч
1	-		
2	3	0	7,167
3	2	0	7,667
4	1	0	5,833

Для следующего шага уточним длительность выполнения каждой задачи закрепленным исполнителем и скорректируем временные показатели. Выпишем задачи, которые можно выполнить на этом шаге, и определим время начала каждой задачи как максимум из времени завершения предыдущей задачи и освобождения исполнителя (Таблица 3.9).

Таблица 3.9 - Время завершения задач каждым из специалистов

Задача	1	2	3	4
4	5,833+7,000	5,833+9,167	5,833+7,833	5,833+7,000
5	5,833+8,000	5,833+8,833	5,833+9,000	5,833+7,833
6	7,667+7,167	7,667+9,000	7,667+8,000	7,667+7,167
7	7,667+9,833	7,667+7,833	7,667+9,000	7,667+8,000
8	7,667+10,000	7,667+8,000	7,667+9,167	7,667+7,833
9	7,167+5,167	7,167+6,000	7,167+8,000	7,167+4,833
10	7,167+5,833	7,167+6,833	7,167+6,833	7,167+5,167

Расставим приоритеты задачам согласно их временному резерву, приведенному в Таблице 3.3 (Таблица 3.10).

Таблица 3.10 - Приоритизация задач по критерию скорейшего завершения

Работа	1	2	3	4	приор
4	12,833	15	13,666	12,833	7
5	13,833	14,666	14,833	13,666	3
6	14,834	16,667	15,667	14,834	6
7	17,5	15,5	16,667	15,667	1
8	17,667	15,667	16,834	15,5	2
9	12,334	13,167	15,167	12	4
10	13	14	14	12,334	5

Выбрав задачи 5,7,8,9, выполним с ними действия, аналогичные тем, что представлены в табл. 3.6, 3.7 и т.д. Далее перейдем к следующим задачам до тех пор, пока не останется ни одной из неназначенных задач. Более подробно все этапы решения представлены в приложении.

Когда все задачи распределены в соответствии с приоритетами и для каждой задачи закреплен исполнитель можно вернуться к подсчету критического пути.

Критический путь, посчитанный по обобщенному среднему времени до приоритизации и распределения, отражен в Таблице 3.11.

Таблица 3.11 - Критический путь, посчитанный по обобщенному среднему времени до приоритизации и распределения

Задача	средняя длит	Ti раннее	Ti позднее	резерв
1	7,042	0	0,748	0,748
2	7,542	0	0	0
3	8,875	0	1,333	1,333
4	7,75	7,042	11,334	4,292
5	8,417	7,042	7,791	0,748
6	7,834	7,542	11,25	3,708
7	8,667	7,542	7,542	0
8	8,75	7,542	7,875	0,332
9	6	8,875	10,208	1,333
10	6,167	8,875	10,458	1,583
11	6,958	15,375	19,084	3,709
12	7,25	16,208	18,792	2,584
13	8,417	16,208	16,208	0
14	8	16,292	16,625	0,332
15	7	23,458	26,042	2,584
16	8,417	24,625	24,625	0

На критическом пути лежат 4 задачи (2-7-13-16), а значение критического пути равно 26,042. Критический путь, который получен в результате проведенных по алгоритму расчетов, представлен в Таблице 3.12.

Таблица 3.12 - Критический путь, полученный после расчетов по алгоритму

Исполнитель	Задача	Длительность	ES	EF	LS	LF	Slack
4	1	5,833	0,00	5,83	2,34	8,17	2,34
3	2	7,667	0,00	7,67	0,00	7,67	0,00
2	3	7,167	0,00	7,17	3,67	10,83	3,67
1	4	7	5,83	12,83	12,83	19,83	7,00
2	5	8,833	5,83	14,67	8,17	17,00	2,34
2	6	9	7,67	16,67	10,83	19,83	3,17
3	7	9	7,67	16,67	8,00	17,00	0,33
1	8	10	7,67	17,67	7,67	17,67	0,00
4	9	4,833	7,17	12,00	12,17	17,00	5,00
3	10	6,833	7,17	14,00	10,83	17,67	3,67
4	11	6	16,67	22,67	19,83	25,83	3,17
3	12	7	16,67	23,67	18,83	25,83	2,17
2	13	7,833	16,67	24,50	17,00	24,83	0,33
1	14	7,167	17,67	24,83	17,67	24,83	0,00
4	15	6	23,67	29,67	25,83	31,83	2,17
4	16	7	24,83	31,83	24,83	31,83	0,00

Критический путь состоит из 4 задач – 2-8-14-16 и составляет 31,83. При значении критического пути, посчитанного по среднему времени, 26,042. Результат сравнения представлен в Таблице 3.13.

Таблица 3.13 - Результат сравнения длин и значений критического пути полученного разными способами.

Вариант решения	Кол-во работ на критическом пути	Список работ на критическом пути	Время	Соотношение
по средним значениям работ	4	2-7-13-16	26,042	100,00%
математический (сбитые приоритеты)	4	2-8-14-16	31,83	122,23%

В результате расчета критического пути по алгоритму получена более точная оценка длительности проекта, которая поможет точнее запланировать требуемое время на выполнение проекта и не выйти за рамки обозначенных в

начале проекта сроков. Подробные расчеты можно посмотреть в приложении Г.

3.4.2. Коррекция

Помимо штатного хода выполнения проекта, в процессе реализации могут возникать внештатные ситуации, оказывающие влияние на сроки и ресурсы. К их числу относятся, например, увольнение либо временная нетрудоспособность сотрудника, назначенного на выполнение определённого стека задач. Аналогично, возможна ситуация обнаружения ошибки в ранее выполненной задаче, что потребует её повторного решения для устранения выявленных недостатков. Кроме того, в ходе приёмочных встреч заказчик может вносить новые требования, дополняющие изначально согласованный стек задач.

Для минимизации вероятности возникновения внезапных дополнительных задач лицо, принимающее решения (ЛПР), до начала работ над проектом должно обеспечить разработку, согласование и подписание технического задания (ТЗ). Данный подход не исключает полностью появления новых задач, однако позволяет вывести их за рамки первоначально согласованного объёма работ и оформить как отдельную категорию — задания на изменения (ЗНИ). Это предотвращает бесконтрольное внесение правок, поскольку качество выполнения задач и их объём изначально закреплены документально.

Далее рассмотри работу алгоритма в случае появления коррекции.

Предположим, что выполнив порученную ему задачу на шаге 3 (таблица 3.13) исполнитель 4 заболел и стал недоступен для выполнения работ. Пересчитаем альтернативное окончание проекта, начиная с шага 4, если доступно только 3 исполнителя – 1, 2 и 3.

Результат шага четыре с учетом коррекции отражен в Таблице 3.14.

Таблица 3.14 - Обновленное закрепление задач за исполнителями на шаге 4 с учетом коррекции.

Исполнитель	Задача	Время начала	Время окончания
1	14	20,83	28,00
2	11	20,83	30,16
3	12	23,33	31,83

Если сравнить время окончания в Таблицах 3.14 с аналогичной таблицей до коррекции в приложении, то видно, что время окончания задачи 14 не изменилось, а для задач 11 и 12 увеличилось в связи с изменением исполнителей из-за произведенной корректировки.

Теперь с учетом отсутствия исполнителя 4 перераспределим задачи 15 и 16 на шаге 5. Результат представлен в Таблице 3.15.

Таблица 3.15 - Обновленное закрепление задач за исполнителями на шаге 5 с учетом коррекции.

Исполнитель	Задача	Время начала	Время окончания
1	16	28	35,83
2	-	-	-
3	15	31,83	38,16

Из таблице выше видим, что обновленное критическое время проекта равно 38,16. Дополним результирующую Таблицу 3.13 новым результатов (Таблица 3.16).

Результаты вычислений с использованием алгоритма, включающего механизм коррекции, закономерно превышают показатели, получаемые без её применения. Такой подход обеспечивает более адекватную оценку фактического времени завершения проекта в условиях выбытия исполнителя или иных изменений, возникающих в процессе его реализации.

Таблица 3.16 - Обновленная таблица сравнения различных вариантов решения задачи

Вариант решения	Кол-во работ на критическом пути	Список работ на критическом пути	Время	Соотношение
по средним значениям работ	4	2-7-13-16	26,042	100,00%
По алгоритму	4	2-8-14-16	31,83	122,23%
По алгоритму с коррекцией	4	2-8-14-16	38,16	146,53%

3.5 Выводы

Разработанный алгоритм обеспечивает более точное определение продолжительности проекта с учётом динамических изменений, возникающих в ходе его реализации. К таким изменениям относятся, в частности, выбытие исполнителя по различным причинам, выявление ошибок на этапе тестирования уже завершённых задач, а также включение в стек работ дополнительных задач, не предусмотренных на стадии первоначального планирования.

Ключевым элементом функционирования алгоритма выступает механизм приоритизации, адаптирующийся к изменениям, что позволяет повысить гибкость управления проектом и его устойчивость к сбоям.

Процесс работы алгоритма предполагает формирование предварительного плана проекта, который впоследствии уточняется на основе фактических оценок исполнителей и анализа текущего прогресса выполнения задач.

ГЛАВА 4. РЕАЛИЗАЦИЯ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ СИСТЕМЫ УПРАВЛЕНИЯ ДЕЯТЕЛЬНОСТЬЮ IT-КОМПАНИИ

В этой главе описана реализация программного комплекса, обеспечивающего автоматизацию оценки общей длительности планируемых проектов и назначения исполнителей для каждой из отдельных задач.

Рассматриваются ключевые аспекты разработки и реализации программного средства, направленного на автоматизацию процесса принятия решений в контексте оптимального распределения задач между исполнителями. Особое внимание уделяется обеспечению завершения проекта в пределах заданных временных рамок с достижением максимально возможного уровня качества исполнения. При этом учитываются ограничения, обусловленные текущей доступностью человеческих ресурсов, их квалификацией и загруженностью. Предлагаемое решение базируется на применении алгоритмических методов, позволяющих учитывать множество факторов, таких как компетенции исполнителей, приоритетность задач и динамические изменения в структуре проекта. В результате данное программное средство способствует повышению эффективности управления деятельностью IT-компании за счет минимизации временных затрат и оптимизации распределения ресурсов при сохранении высокого качества исполнения поставленных задач.

Первая часть главы посвящена проектированию структуры базы данных. Структура программного обеспечения для управления деятельностью IT-компании представлена во второй части главы. В третьей части главы приведены результаты функционирования разработанного приложения «TaskFlow – управление задачами», помогающего осуществлять управление в IT-компании путем оценки длительности всего IT-проекта и отдельных его задач в зависимости от назначенных исполнителей. Четвертая часть посвящена экспериментальному подтверждению эффективности

разработанных моделей и алгоритмов. Пятая часть содержит выводы по главе.

4.1 Проектирование структуры базы данных

Разработаем структуру базы данных для реализации программного комплекса обеспечивающего автоматизацию оценки общей длительности планируемых проектов и назначения исполнителей для каждой из отдельных задач. Данная база данных является ключевым элементом информационной системы, обеспечивающим надежное хранение, обработку и управление данными [28, 71].

На первом этапе разработки осуществляется построение логической структуры (модели) данных. Логическая модель данных представляет собой формализованное описание предметной области, включающее определение основных сущностей, их атрибутов и взаимосвязей между ними. Для корректного формирования модели необходимо выделить ключевые объекты, такие как проекты, задачи, исполнители, временные параметры выполнения и другие характеристики, влияющие на процесс управления проектами [31, 71]. В общем виде эти требования можно сгруппировать следующим образом (Рисунок 4.1 и Рисунок 4.2).

Одной из ключевых сущностей, необходимых для решения поставленной задачи, является проект. Проект представляет собой основной объект предметной области, который характеризуется не только набором входящих в него задач, но и рядом дополнительных параметров, обеспечивающих его идентификацию и описание. К таким параметрам относятся: наименование проекта, его описание, даты начала и окончания, а также расчетная продолжительность выполнения.

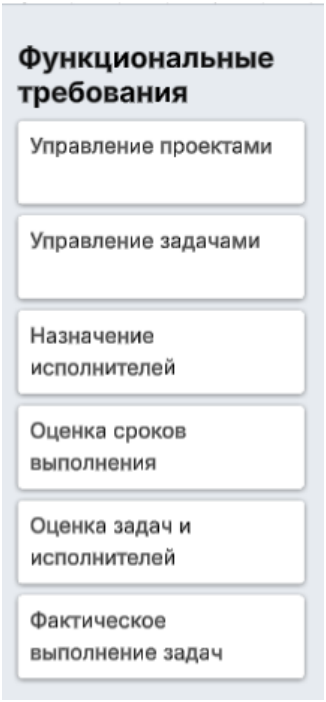


Рисунок 4.1 – Функциональные требования

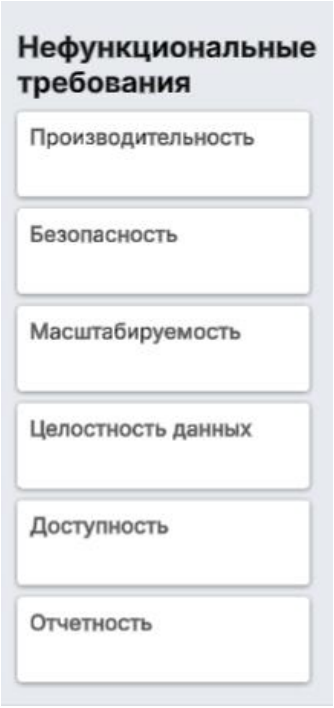


Рисунок 4.2 – Нефункциональные требования

Таким образом, сущность "Проект" может быть описана следующими атрибутами:

– идентификатор проекта (id)– первичный ключ,тип данных serial.

– название (наименование) проекта (name)– тип данных: varchar(255), содержит название данного проекта

– описание проекта (description) – текстовое поле содержащее описание данного проекта

– текущий статус проекта (project_status) - (например, "в процессе", "завершено"), тип данных: varchar(50).

– дата начала проекта (start_date) – тип данных date, содержит дату начала (старта) проекта

– дата окончания проекта (end_date)– тип данных date, содержит дату окончания проекта

– расчетная продолжительность проекта (estimated_duration) – тип данных interval, может быть рассчитано автоматически

Поскольку проект состоит из отдельных задач (работ), которые выполняются в рамках его реализации, следующей ключевой сущностью является задача. Данная сущность предназначена для хранения информации о задачах, связанных с конкретными проектами, и играет центральную роль в процессе управления проектами.

Каждая задача характеризуется рядом атрибутов, обеспечивающих её уникальную идентификацию и описание. Основные атрибуты сущности "Задача" включают:

– уникальный идентификатор задачи (id) – первичный ключ,тип данных serial.

– идентификатор проекта, к которому относится задача (project_id) - тип данных: int.

– название (наименование) задачи (name) - тип данных: varchar(255), содержит наименование задачи

– описание задачи (description) - тип данных: text.

– приоритет задачи (priority) - тип данных: int.

– текущий статус задачи (task_status) - (например, "в процессе", "завершено"), тип данных: varchar(50).

– минимальная оценка времени выполнения задачи (optimistic_time) - тип данных: interval.

– наиболее вероятное время выполнения задачи (most_likely_time) - тип данных: interval.

– максимальная оценка времени выполнения задачи (pessimistic_time) - тип данных: interval.

– ожидаемое время выполнения задачи (expected_duration) - тип данных: interval. рассчитывается автоматически как (optimistic_time + 4 * most_likely_time + pessimistic_time) / 6.

– дата начала задачи (start_date) - тип данных: date, содержит дату начала (старта) задачи

– дата окончания задачи (end_date) - тип данных: date, содержит дату окончания задачи

Поскольку один проект может содержать несколько задач, между сущностями "Проект" и "Задача" устанавливается связь типа "один ко многим" . Это означает, что одна запись в таблице "Проект" может быть связана с несколькими записями в таблице "Задача" , но каждая задача относится только к одному проекту [35]. Такая организация данных позволяет адекватно отразить иерархическую структуру проектов и входящих в них задач, обеспечивая целостность и логичность модели.

Для дальнейшего развития модели введем сущность "Исполнитель" , которая предназначена для хранения информации об исполнителях задач. Данная сущность играет ключевую роль в процессе распределения задач между участниками проекта и управления их выполнением. Каждый исполнитель характеризуется рядом атрибутов, обеспечивающих его уникальную идентификацию и описание. Основные атрибуты сущности "Исполнитель" включают:

– уникальный идентификатор исполнителя (id) - тип данных: serial.

– имя исполнителя (name) - тип данных: varchar(255), содержит фио исполнителя

– электронная почта исполнителя (email) - тип данных: varchar(255), содержит адрес электронной почты исполнителя

– роль исполнителя (role) - тип данных: varchar(50), содержит название роли исполнителя, например, "разработчик", "менеджер"

Учитывая, что в рамках проекта задачи могут выполняться не только последовательно, но и параллельно или параллельно-последовательно, возникает необходимость введения дополнительной сущности "Назначение задачи". Данная сущность предназначена для отражения информации о распределении задач между исполнителями и обеспечения учета их текущего статуса выполнения.

Сущность "Назначение задачи" выступает связующим звеном между сущностями "Задача" и "Исполнитель" , позволяя учитывать сложные сценарии выполнения задач, такие как параллельное или параллельно-последовательное выполнение. Это особенно важно для автоматизации процесса назначения задач и контроля их выполнения в условиях многозадачности. Основные атрибуты сущности "Назначение задачи" включают:

– уникальный идентификатор записи(id) - тип данных: serial.

– идентификатор задачи (task_id) -тип данных: int.

– идентификатор исполнителя (user_id) - тип данных: int.

– дата назначения задачи (assigned_date) - тип данных: date.

В связи с тем, что начало выполнения одних задач в проекте может зависеть от завершения других задач, возникает необходимость учета таких зависимостей. Для этого вводится сущность "Зависимость задач" , которая

предназначена для хранения информации о взаимосвязях между задачами и обеспечения корректного управления их выполнением.

Сущность "Зависимость задач" играет ключевую роль в модели данных, так как позволяет учитывать логические и временные ограничения, накладываемые порядком выполнения задач. Это особенно важно для автоматизации процессов планирования и контроля выполнения проектов, где задачи могут быть связаны сложными зависимостями (например, последовательными или параллельно-последовательными связями).

Основные атрибуты сущности "Зависимость задач" включают:

– уникальный идентификатор зависимости (id) - тип данных: serial.

– идентификатор предшествующей задачи (predecessor_task_id) - тип данных: int.

– идентификатор последующей задачи (successor_task_id) - тип данных: int.

В рамках разработанной базы данных вводится сущность "Критерии оценки", которая предназначена для хранения информации о критериях, используемых для оценки задач и исполнителей. Данная сущность играет важную роль в обеспечении объективности и прозрачности процессов оценки, позволяя формализовать параметры, по которым выполняется анализ эффективности как отдельных задач, так и работы исполнителей.

Основные атрибуты сущности "Критерии оценки" включают:

– уникальный идентификатор критерия (id) - тип данных: serial.

– название критерия (name) - тип данных: varchar(255).

– описание критерия (description) - тип данных: text.

Сущность «Оценка задач по критериям» хранит оценки задач по различным критериям и содержит атрибуты:

– уникальный идентификатор записи (id) - тип данных: serial.

– идентификатор задачи (task_id) - тип данных: int.

– идентификатор критерия (criteria_id) - тип данных: int.

– оценка по критерию (score) - тип данных: numeric.

Сущность «Оценка исполнителей по критериям» хранит оценки задач по различным критериям и содержит атрибуты:

– уникальный идентификатор записи (id) - тип данных: serial.

– идентификатор исполнителя (user_id) - тип данных: int.

– идентификатор критерия (criteria_id) - тип данных: int.

– средняя оценка исполнителя по критерию (average_score) - тип данных: numeric.

Одной из важнейших сущностей данной модели будет являться сущность «Расписание». Именно в ней должны отражаться результаты решения оптимизационной задачи с учетом всех необходимых ограничений. Каждая отдельная запись должна содержать информацию о запланированном и фактическом времени начала каждой работы. С учетом того, что интерес представляют лишь работы, выполняемые для определенных проектов, то связана эта сущность будет с сущностью «Работы проекта» по двум ее ключевым полям: идентификатор проекта и идентификатор работы. Кроме того, целесообразно хранить информацию о сотруднике, который выполнял эту работу. В связи с этим, сущность «Расписание» будет включать в себя следующие поля:

– идентификатор расписания (первичный ключ, целое положительное число);

– идентификатор работы (внешний ключ);

– идентификатор проекта (внешний ключ);

– время начала планируемое;

– время начала фактическое.

В виду случайного характера длительности выполнения работ, плановое время ее окончания не всегда будет совпадать с фактическим. Здесь возможны две ситуации:

– плановое время меньше фактического;

– плановое время превышает фактическое.

В первом случае для выполнения проекта это скорее преимущество, чем недостаток. Однако, при систематическом отклонении времени выполнения одной и той же работы в одну и ту же сторону возможно возникнет необходимость в корректировке ее параметров. В случае же превышения планового времени может к тому же возникнуть ситуация, требующая оперативной корректировки графика. В связи с этим, в базе данных необходима сущность «Несвоевременное завершение», которая будет фиксировать любые отклонения фактических и плановых сроков для работ. Эта сущность будет связана с сущностью «Расписание» и, при необходимости, должна обеспечивать возможность определения не только работы, которая была задержана или выполнена с опережением, но и сотрудника, выполнявшего ее. В связи с этим, она должна содержать следующие атрибуты:

– идентификатор (первичный ключ, целое положительное число);

– идентификатор расписания (внешний ключ);

– идентификатор причины (внешний ключ).

Как видно из данного списка, кроме самого факта фиксации отклонения от планового срока той или иной работы целесообразно определить причину ее несвоевременного выполнения. Это необходимо для анализа факторов, влияющих на длительность той или иной работы, а также учет той или иной ситуации при коррекции параметров работ. Справочник причин несвоевременного завершения будет включать в себя лишь два атрибута:

– идентификатор причины (первичный ключ, целое положительное число);

– наименование причины.

На основании описанных выше сущностей и связей между ними, спроектируем структуру базы данных. [34]

Ниже представлена логическая модель базы данных (Рисунок 4.3):

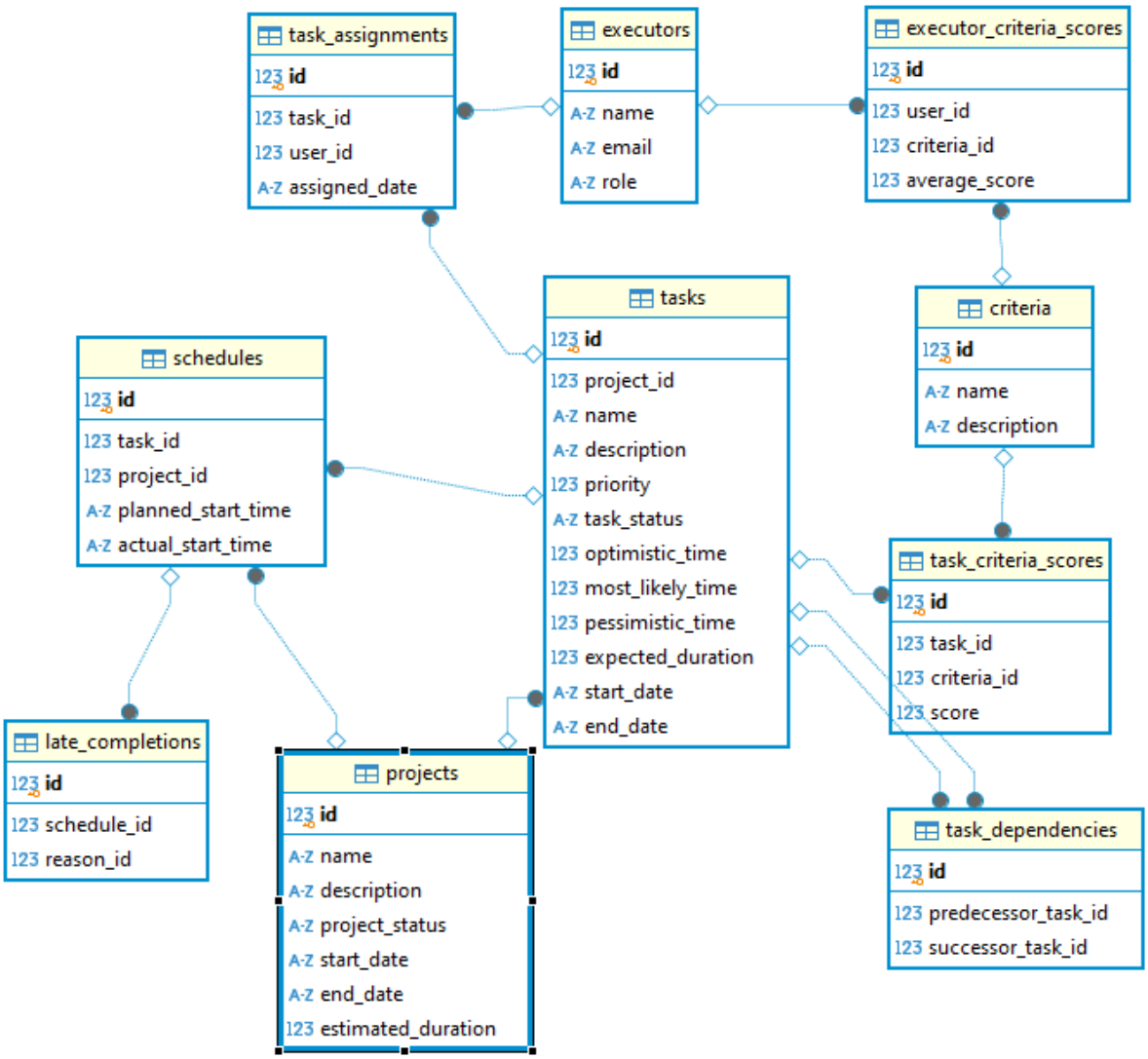


Рисунок 4.3 – Логическая модель базы данных

4.2 Структура программного обеспечения для управления деятельностью IT-компании

Рассмотрим реализацию описанной выше структуры проекта и базы данных для решения задач в IT-проектах.

Программный комплекс представляет собой систему управления деятельностью IT-компании за счет автоматизации и оптимизации процессов планирования, контроля, оценки и коррекции выполнения проектных задач.

В основе программного комплекса будут лежать множество моделей и алгоритмов, разработанных в главе 2 и 3 соответственно.

Программный комплекс должен обеспечивать следующие основные возможности:

- позволять управлять процессом назначения исполнителей, определяя наилучшего исполнителя каждой из задач;

- позволять управлять графиком IT-проекта, назначая каждой задаче наилучшее время;

- прогнозировать сроки IT-проекта и возможные риски его несвоевременного завершения;

- вносить оперативные корректировки в принятые решения из-за появления новых задач/исполнителей, изменения их длительностей и т.д.

Исходя из данного анализа, можно прийти к следующим выводам:

- в проектируемом программном средстве необходимо предусмотреть подсистему планирования, позволяющую не только определять время начала выполнения каждой из задач, но и назначать им исполнителей;

- необходимо предусмотреть подсистему оценивания, позволяющую учитывать квалификацию исполнителя, решающего задачу, и предусматривающую коррекцию временных показателей по самым разнообразным причинам;

- поскольку подразумевается итеративная работа по планированию задач проекта, предусмотрена подсистема коррекции, которая фиксирует статус каждой задачи после каких-либо согласований (с руководством или заказчиком) и корректирует график на ближайший период, исходя из текущей ситуации [72].

Для выполнения данных требований при реализации программного обеспечения для управления деятельностью IT-компании была разработана его структура, представленная на Рисунке 4.4.

Рассмотрим необходимость функционирования каждой из подсистем программного комплекса. Основу составляет подсистема планирования,

которая определяет множество задач, которые будут выполнены в данный слот времени, время начала и окончания этих задач и соответствующих им исполнителей.

Рисунок 4.4 – Структура программного комплекса

В данном случае предполагается, что, находясь в постоянном взаимодействии с подсистемой оценивания и имея возможность получить ориентировочное время решения каждой задачи каждым из сотрудников, система предоставит решение многокритериальной оптимизационной задачи,

связанной с распределением задач по исполнителям и оценке их времени начала. Кроме того, из множества задач требуется отобрать те, которые будут начаты в данный временной интервал, т.е. некоторым образом приоритизировать задачи. В случае, когда планирование выполнено до конца, это можно сделать с помощью временных резервов. В противном случае требуется предложить свой подход к приоритизации. В подсистеме предусмотрена функция предварительного планирования, которая назначит каждой задаче время начала (в зависимости от исполнителя) и оптимизация, которая поможет выбрать наилучшее время и наилучшего исполнителя. Поскольку предполагается возможность оперирования с множеством критериев, предусмотрена процедура свертки критериев и выбора наилучшего исполнителя для каждой задачи. Алгоритмы, на основании которых реализуется планирование и назначение, подробно описаны в главе 3.

Основной задачей подсистемы оценивания является получение оценки решения каждой из задач каждым исполнителем с учетом разнообразных стохастических факторов (опыта исполнителя для данной задачи, возможности внесения корректировок заказчиком и т.д.). Математический аппарат, который используется для оценивания, подробно описан в третьей части главы 2.

С учетом того, что некоторые задачи будут завершены несвоевременно (что связанно со случайным характером времени выполнения работ и возможных изменений), в структуре предусмотрена подсистема коррекции, которая, в соответствии с новыми сроками по задачам и актуальным расписанием работы исполнителей корректирует имеющийся план-график.

Одной из основных особенностей IT-проектов является большая степень неопределенности при оценки сроков выполнения как отдельных задач так и всего проекта в целом. Как отмечалось ранее, новые требования, озвучиваемые заказчиком после согласования определенной части задач, появление или болезнь новых исполнителей и ряд других событий вносит

большую неопределенность в сроки проекта. В связи с этим, целесообразным является использование системы имитационного моделирования, предусматривающей данные факторы и позволяющей определить не только сроки, но и разнообразные временные риски на основе оценок вероятностных характеристик проекта.

Исполнители взаимодействуют с системой, в основном, путем фиксации своих задач и изменения их статуса. Team Lead как пользователь анализирует план для своей команды, при необходимости меняет исполнителей для задач вручную и также отслеживает процесс выполнения задач исполнителями, входящими в его команду. ЛПР – это руководитель на более высоком уровне (руководитель подразделения или компании в целом, который принимает решения для проектов в целом).

Таким образом, разработана структура программного обеспечения системы управления деятельностью IT-компании, отличающаяся комплексной интеграцией подсистем имитационного моделирования, планирования, оптимизации и базы данных, и обеспечивающая автоматизацию процесса управления IT-компанией путем реализации разработанных алгоритмов.

4.3 Результаты функционирования разработанной системы управления

Рассмотрим функциональные возможности и интерфейс разработанного приложения.

Стартовое окно «Дашборд» приложения «TaskFlow – управление задачами», представлено на рисунке 4.5.

В левой части окна находится список вкладок доступных пользователю: «Дашборд», «Аналитика», «Отчеты», «Проекты», «Задачи», «Исполнители», «Канбан Доска», «Диаграмма Ганта», «Календарь», «Профиль пользователя». Вкладка «Дашбор» является стартовой вкладкой при открытии программы. В центральной части вкладки отражена ключевая информация по количеству проектов и задач, распределению задач, загрузки

сотрудников по ролям, количестве перегруженных сотрудников, срочных «горящих» задачах, если таковые имеются. Если параллельно есть срочные задачи и свободные сотрудники это будет отражено в окне рекомендации.

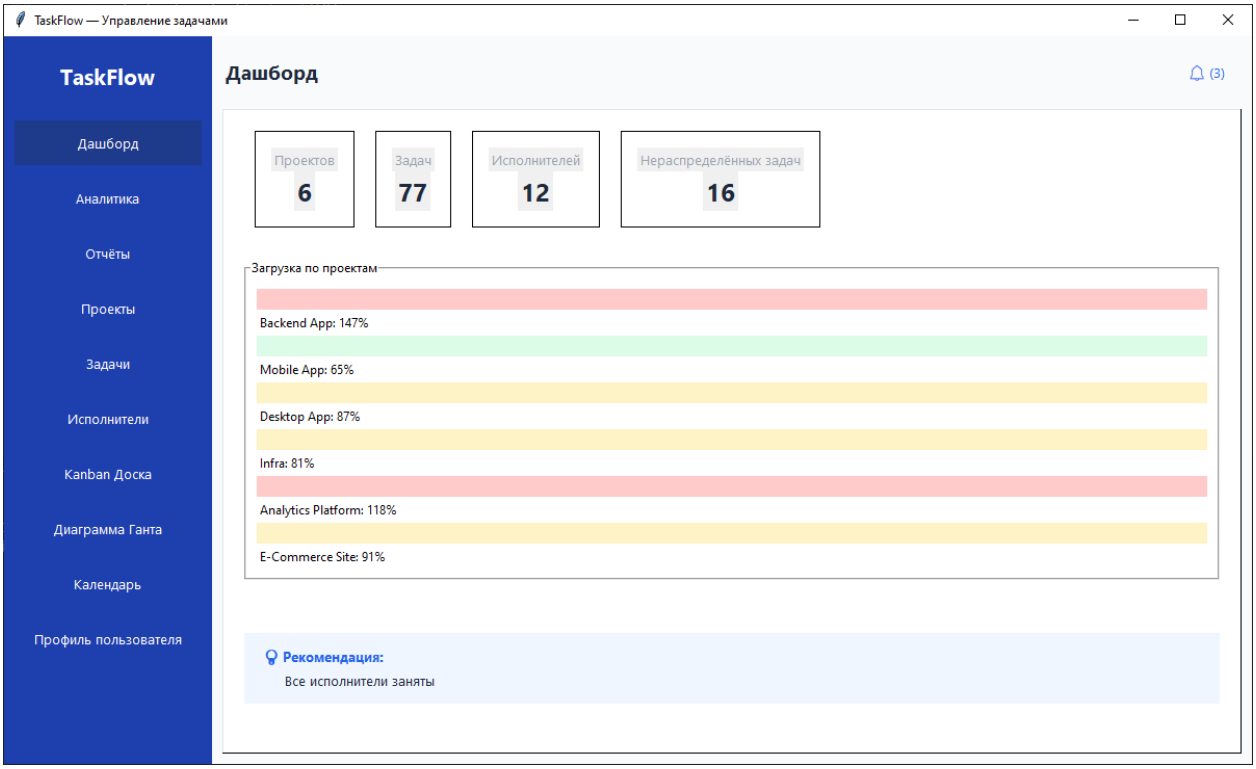


Рисунок 4.5 – Начальная вкладка «Дашборд»

Данная вкладка является ключевым элементом пользовательского интерфейса и предназначено для обеспечения быстрого обзора текущего состояния проектов, задач и исполнителей. Она предоставляет компактную и наглядную информацию, необходимую для эффективного управления проектами.

Следующим ключевым элементом приложения является вкладка «Аналитика», представлена на Рисунке 4.6. На ней представлены графики загрузки команды, прогресса по проектам, а так же отражено среднее время выполнения задач для разных направлений.

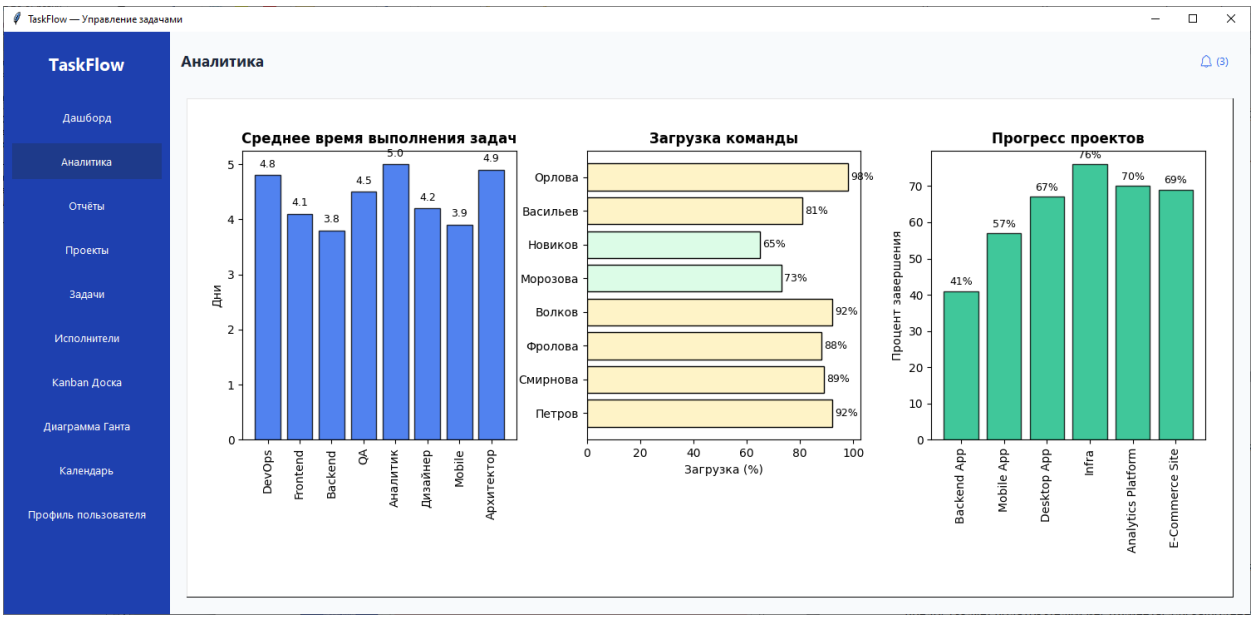


Рисунок 4.6 – Вкладка «Аналитика»

После нее идет вкладка «Отчеты». На этой вкладке представлены стандартные шаблоны отчетов, которые можно быстро использовать с данными в программе. Представлены следующие отчеты:

– еженедельный отчет по проектам;
– отчет по производительности команды;
– отчет о рисках;
– отчет по просроченным задачам;
– отчет по загрузке ресурсов.

Следующим ключевым элементом приложения является вкладка «Проекты», предназначенная для управления проектами. Внешний вид вкладки представлен на Рисунке 4.7.

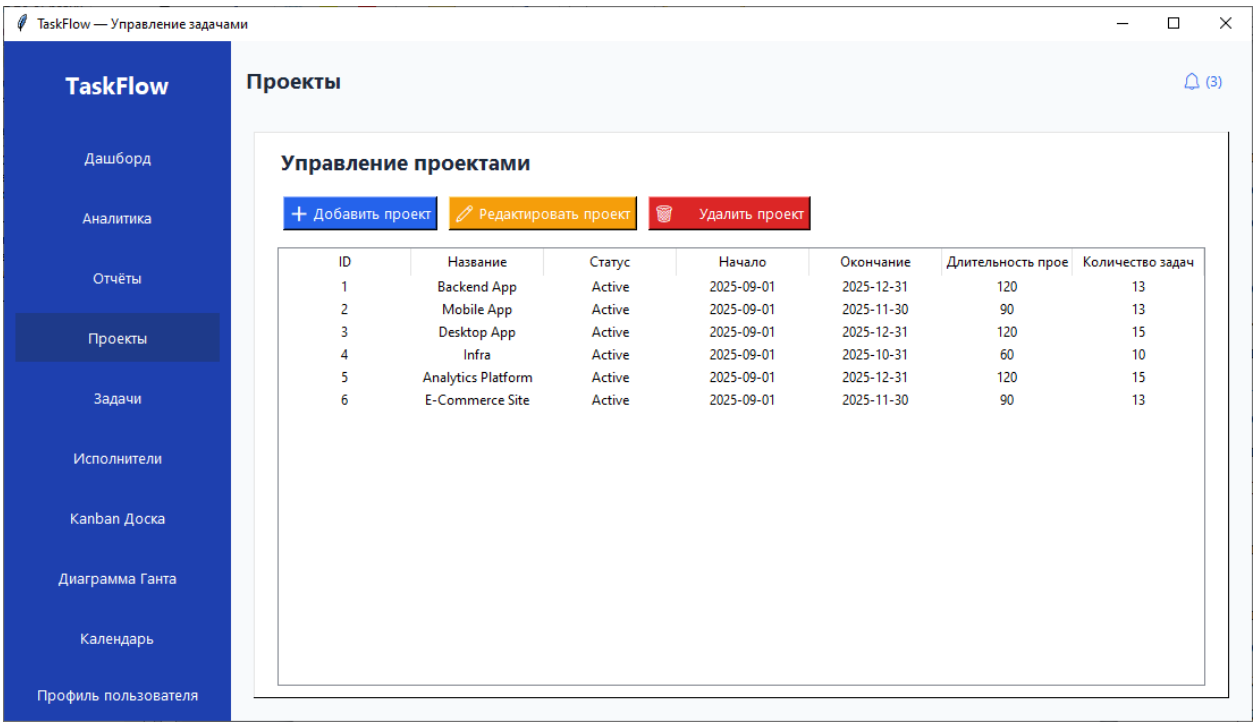
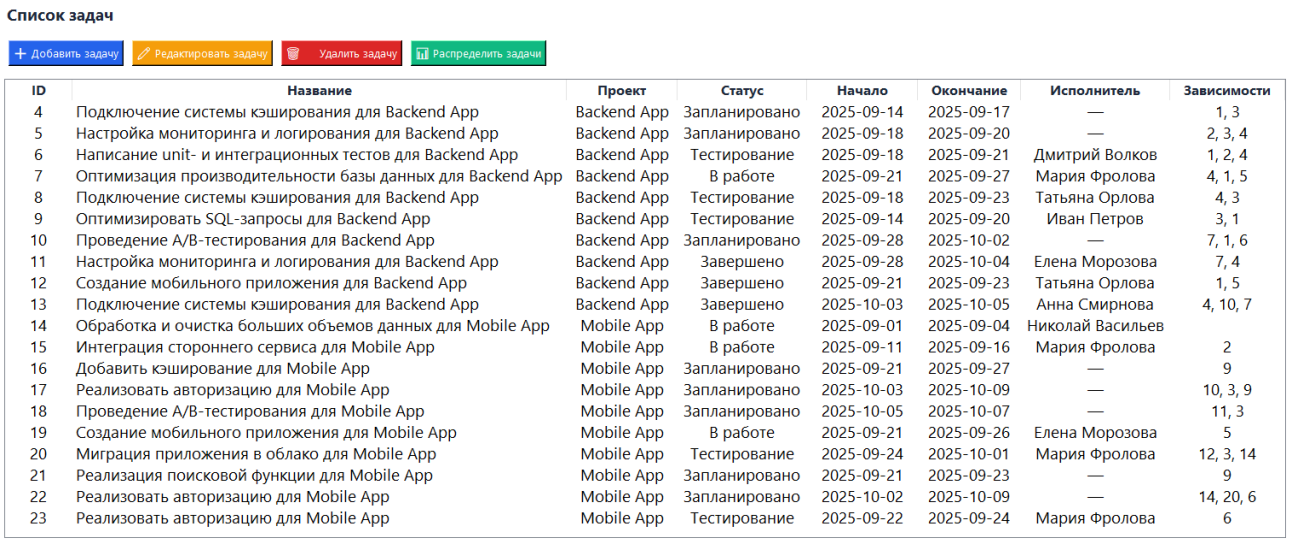


Рисунок 4.7 - Внешний вид вкладки «Проекты»

Данная вкладка предоставляет пользователям инструменты для выполнения таких операций, как добавление, редактирование и удаление проектов, а также обеспечивает удобный доступ к информации о каждом проекте. При нажатии кнопки «Добавить проект» открывается окно добавления проекта, где необходимо ввести наименование проекта, его описание. Проекту автоматически будет выставлен статус, текущая дата начала, которую при необходимости, можно поменять, и доступна кнопка расчета даты окончания проекта.

Вкладка «Задачи» предназначена для управления задачами внутри проектов и предоставляет инструменты для их анализа и контроля. Внешний вид вкладки представлен на Рисунке 4.8.

На вкладке отображается таблица всех задач, содержащая основную информацию, описанную в главе 2: идентификатор, название, проект, статус, исполнитель (если задача распределена), даты начала и окончания работ по данной задаче, а также связи с предшествующими и последующими задачами.



ID	Название	Проект	Статус	Начало	Окончание	Исполнитель	Зависимости
4	Подключение системы кэширования для Backend App	Backend App	Запланировано	2025-09-14	2025-09-17	—	1, 3
5	Настройка мониторинга и логирования для Backend App	Backend App	Запланировано	2025-09-18	2025-09-20	—	2, 3, 4
6	Написание unit- и интеграционных тестов для Backend App	Backend App	Тестирование	2025-09-18	2025-09-21	Дмитрий Волков	1, 2, 4
7	Оптимизация производительности базы данных для Backend App	Backend App	В работе	2025-09-21	2025-09-27	Мария Фролова	4, 1, 5
8	Подключение системы кэширования для Backend App	Backend App	Тестирование	2025-09-18	2025-09-23	Татьяна Орлова	4, 3
9	Оптимизировать SQL-запросы для Backend App	Backend App	Тестирование	2025-09-14	2025-09-20	Иван Петров	3, 1
10	Проведение A/B-тестирования для Backend App	Backend App	Запланировано	2025-09-28	2025-10-02	—	7, 1, 6
11	Настройка мониторинга и логирования для Backend App	Backend App	Завершено	2025-09-28	2025-10-04	Елена Морозова	7, 4
12	Создание мобильного приложения для Backend App	Backend App	Завершено	2025-09-21	2025-09-23	Татьяна Орлова	1, 5
13	Подключение системы кэширования для Backend App	Backend App	Завершено	2025-10-03	2025-10-05	Анна Смирнова	4, 10, 7
14	Обработка и очистка больших объемов данных для Mobile App	Mobile App	В работе	2025-09-01	2025-09-04	Николай Васильев	
15	Интеграция стороннего сервиса для Mobile App	Mobile App	В работе	2025-09-11	2025-09-16	Мария Фролова	2
16	Добавить кэширование для Mobile App	Mobile App	Запланировано	2025-09-21	2025-09-27	—	9
17	Реализовать авторизацию для Mobile App	Mobile App	Запланировано	2025-10-03	2025-10-09	—	10, 3, 9
18	Проведение A/B-тестирования для Mobile App	Mobile App	Запланировано	2025-10-05	2025-10-07	—	11, 3
19	Создание мобильного приложения для Mobile App	Mobile App	В работе	2025-09-21	2025-09-26	Елена Морозова	5
20	Миграция приложения в облако для Mobile App	Mobile App	Тестирование	2025-09-24	2025-10-01	Мария Фролова	12, 3, 14
21	Реализация поисковой функции для Mobile App	Mobile App	Запланировано	2025-09-21	2025-09-23	—	9
22	Реализовать авторизацию для Mobile App	Mobile App	Запланировано	2025-10-02	2025-10-09	—	14, 20, 6
23	Реализовать авторизацию для Mobile App	Mobile App	Тестирование	2025-09-22	2025-09-24	Мария Фролова	6

Рисунок 4.8. – Определение сроков задач и назначение им исполнителей – общий вид вкладки «Задачи»

Дополнительную информацию, такую как, тип задачи, приоритет, дедлайн можно посмотреть в карточке задачи, которая открывается при двойном клике на задачу (Рисунок 4.9).

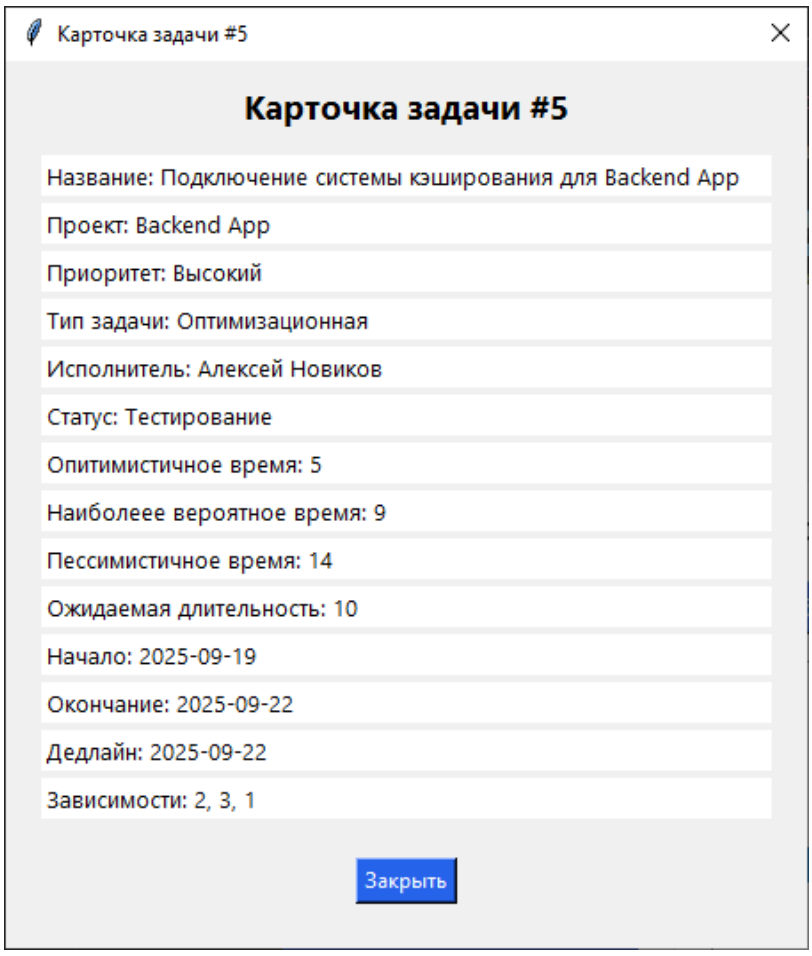


Рисунок 4.9 – Определение сроков задач и назначение им исполнителей – внешний вид карточки отдельной задачи

По кнопке «Добавить задачу» открывается окно добавления задачи, в котором можно записать всю информацию, отраженную на Рисунке 4.9.

Окно редактирования задачи выглядит аналогичным образом, но заполнено информацией добавленной при создании задачи или обновленной во время ее распределения.

По кнопке «Распределить задачи» можно перейти в окно распределения задач. Внешний вид окна представлен на Рисунке 4.10.

В данном окне есть возможность выбрать в правой части окна, как отдельные задачи для распределения, так и все задачи, входящие в какой либо проект (для этого надо выбрать пункт в проекте «Выбрать все»).

В левой части окна можно выбрать исполнителей для выполнения данных задач, в списке находятся только те исполнители, которые доступны для распределения в данный момент.

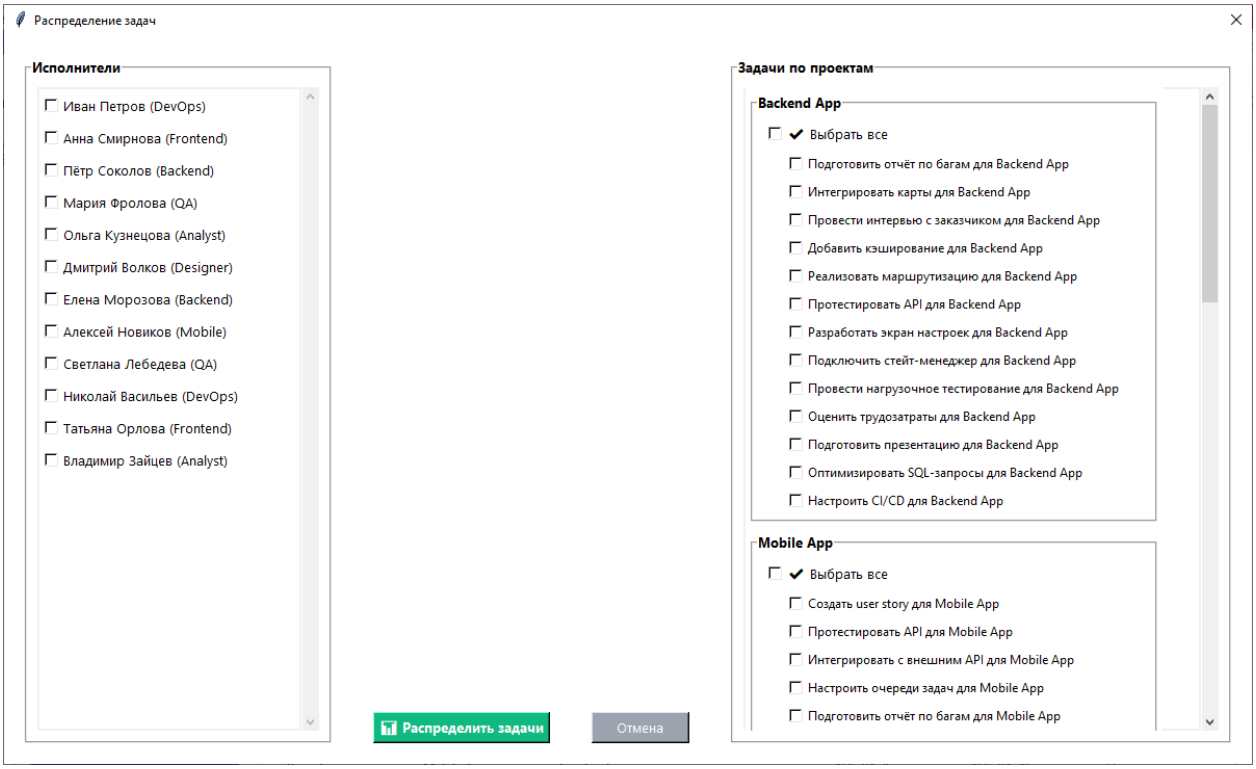


Рисунок 4.10 – Внешний вид окна «Распределение задач»

Это окно позволяет, в том числе, перераспределить задачи на других исполнителей при выбытии (отсутствии) одного из уже назначенных ранее исполнителей. При успешном распределении задач появится окно представленное на Рисунке 4.11.

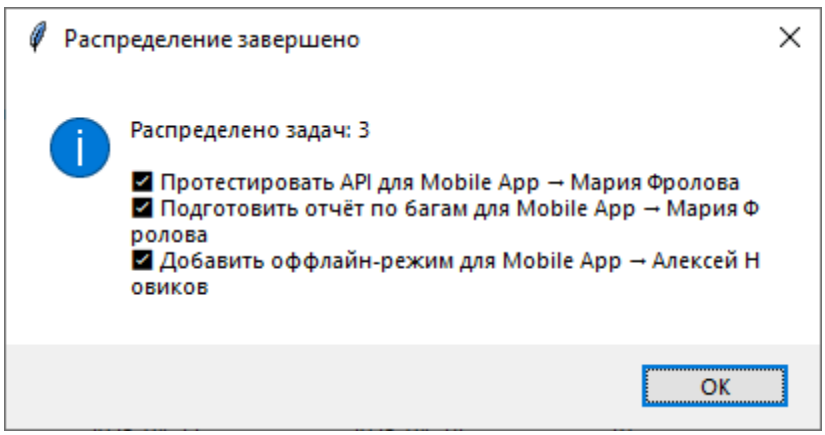


Рисунок 4.11 – Окно «Распределение завершено»

Если не были выбраны исполнители для назначения или задачи для распределения, при нажатии на кнопку «Распределить задачи», появится окно с сообщением об ошибке.

Вкладка «Исполнители» предназначена для управления исполнителями и их оценки.

На вкладке отображается таблица всех исполнителей, Рисунок 4.12.

В таблице содержится основная информация об исполнителе: ФИО, роль, статус на момент просмотра, стаж и контактные данные. Кроме основных кнопок «Добавить», «Редактировать», «Удалить» исполнителя на вкладке имеется дополнительная кнопка «Критерии», при нажатии на которую открывается окно «Критерии оценки исполнителей» содержащее различные критерии и оценку каждого исполнителя по ним. Можно просмотреть как все критерии по всем исполнителям, так и выбрать конкретный критерий.

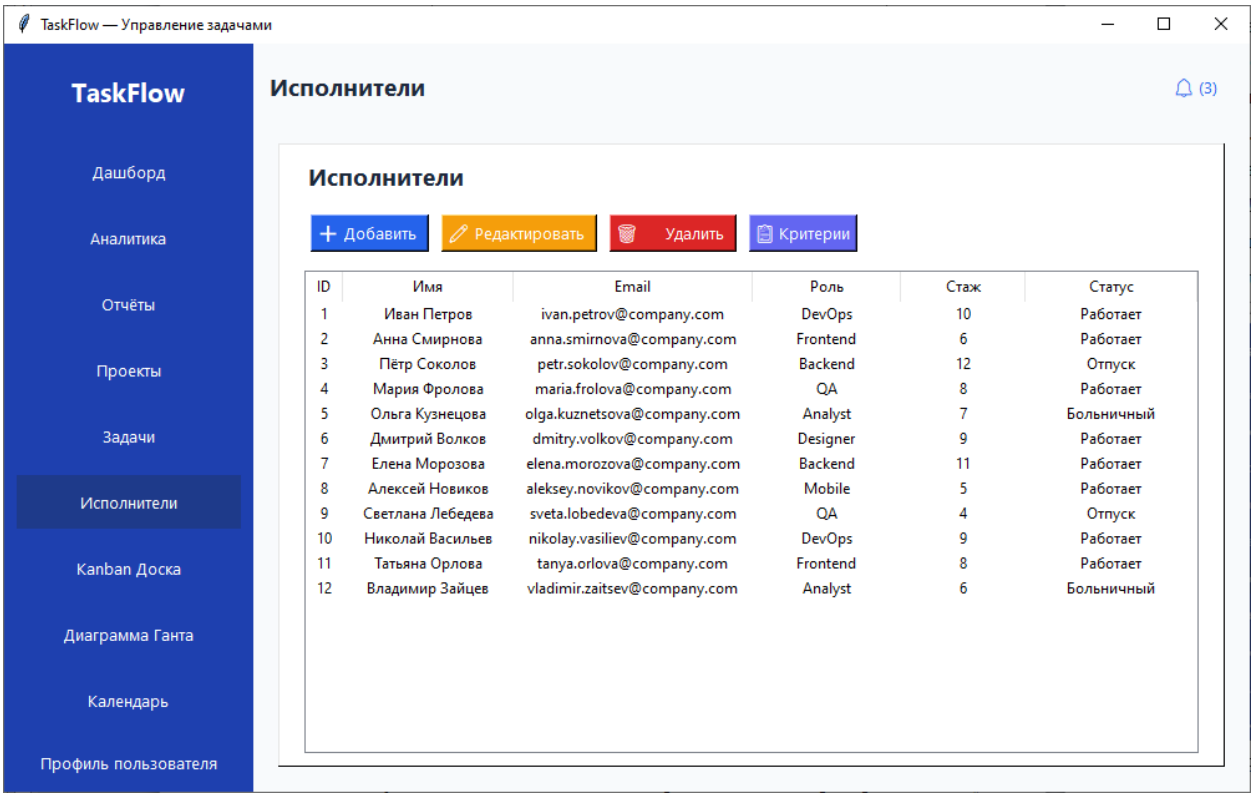


Рисунок 4.12 – Внешний вид вкладки «Исполнители»

Например, владение языком Python, Рисунок 4.13.

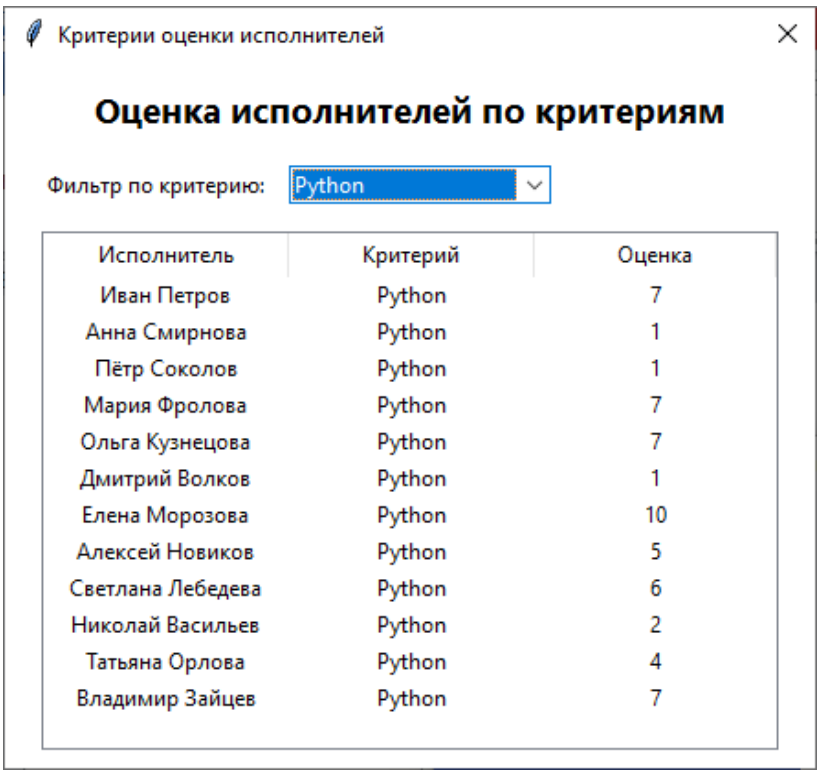


Рисунок 4.13 – Окно «Критерии оценки исполнителей»

Следующая вкладка – «Kanban Доска». На которой представлены задачи распределенные в колонки по актуальным статусам (Рисунок 4.14).

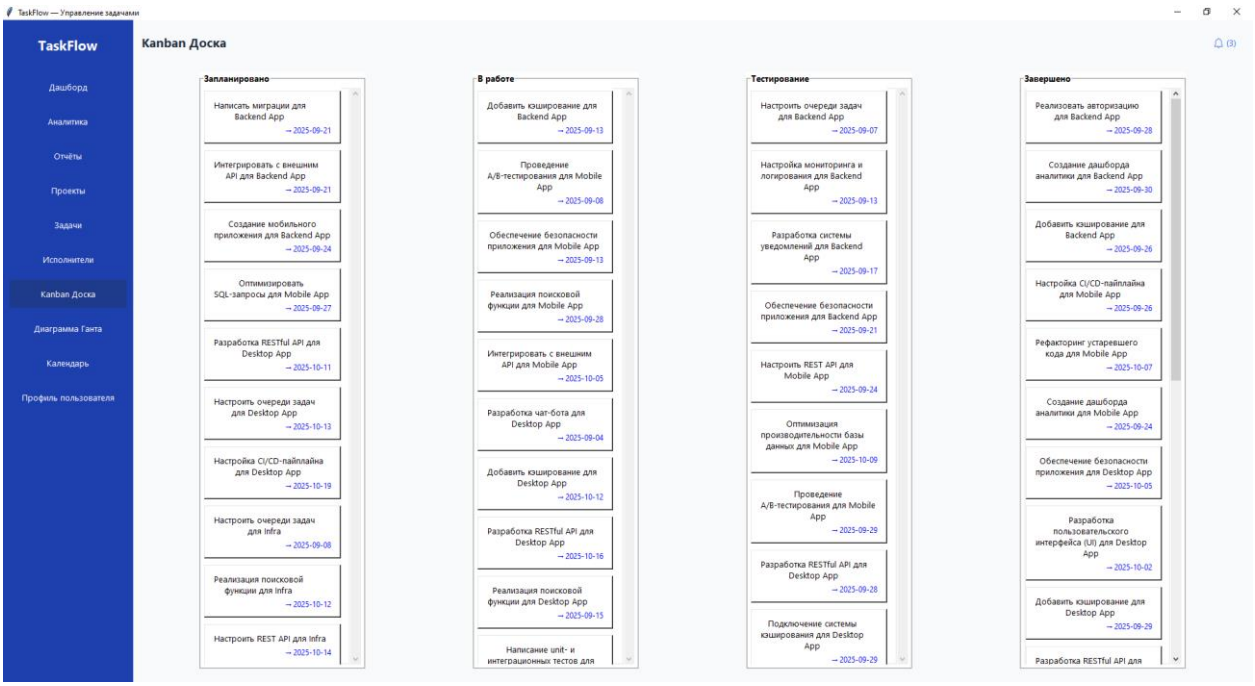


Рисунок 4.14 - «Kanban Доска»

Затем расположена вкладка «Диаграмма Ганта», рисунок 4.15.

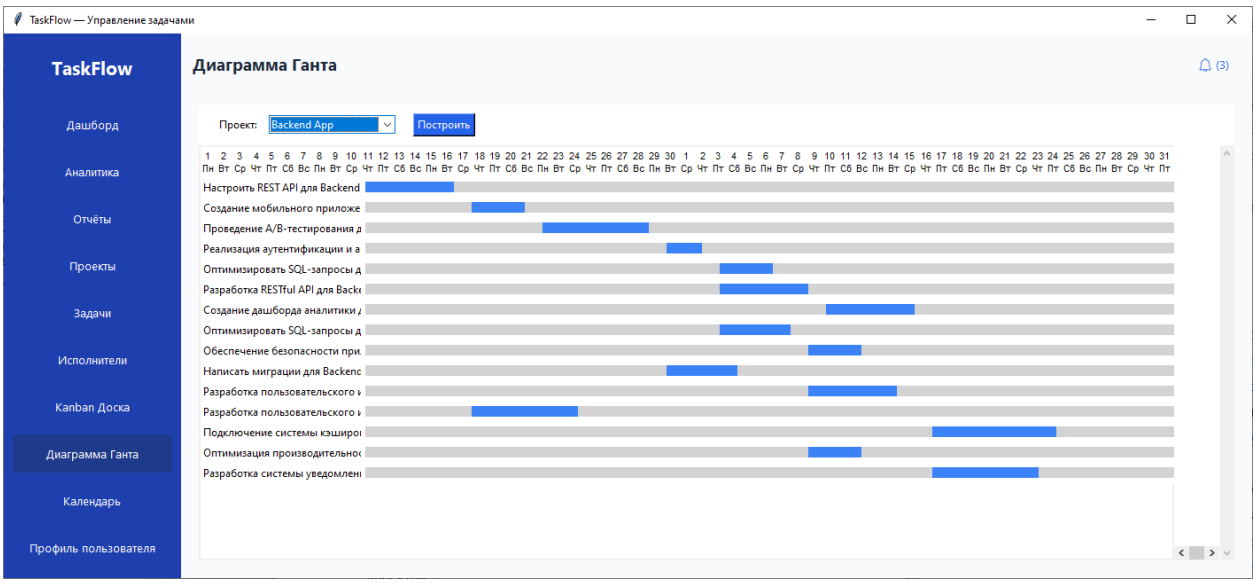


Рисунок 4.15 – «Диаграмма Ганта»

На диаграмме Ганта можно посмотреть и проанализировать сроки выполнения задач, их последовательность, а так же взаимосвязь между ними. Кроме того она позволяет оценить общий ход проекта во времени, длительность и сроки выполнения каждого этапа, если проект делится на этапы. На представленной диаграмме можно посмотреть как отдельные проекты, так и все задачи находящиеся в работе вне зависимости от проектов.

Далее предусмотрена вкладка «Календарь», при нажатии на которую открывается окно «Календарь загрузки исполнителей». Эта вкладка предназначено для визуализации загрузки исполнителей в рамках проектов, рисунок 4.16.

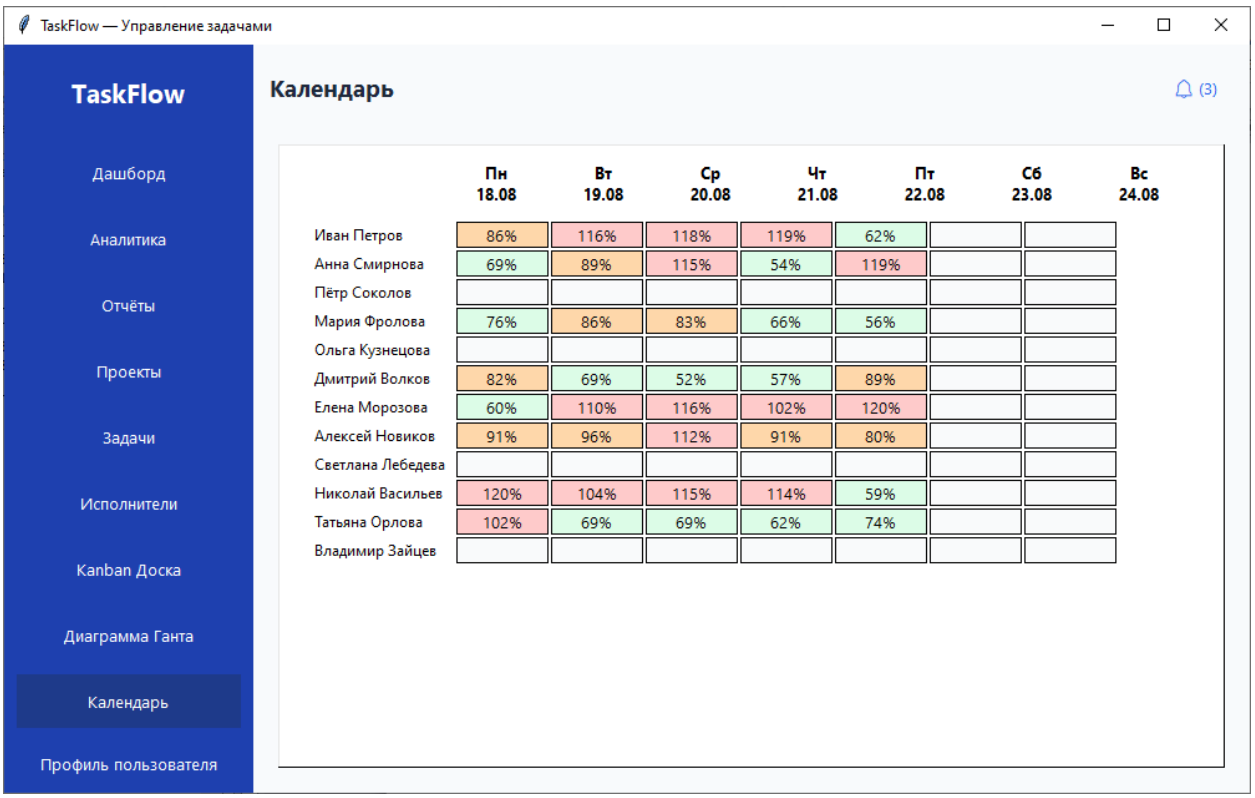


Рисунок 4.16 – Вкладка «Календарь»

Окно содержит календарь на неделю с отображением загрузки каждого исполнителя с использованием цветовой маркировки для обозначения уровня загрузки: нормальная, высокая или перегрузка. Такая визуализация позволяет быстро выявить периоды повышенной нагрузки или простоя.

Последней вкладкой программы является вкладка «Профиль пользователя» в которой содержится информация о пользователе, который в данный момент работает в системе. А так же данные о его активности в последнее время. Кроме того на данной вкладке можно сменить цвет темы приложения и включить уведомления. Внешний вид вкладки представлен на рисунке 4.17.

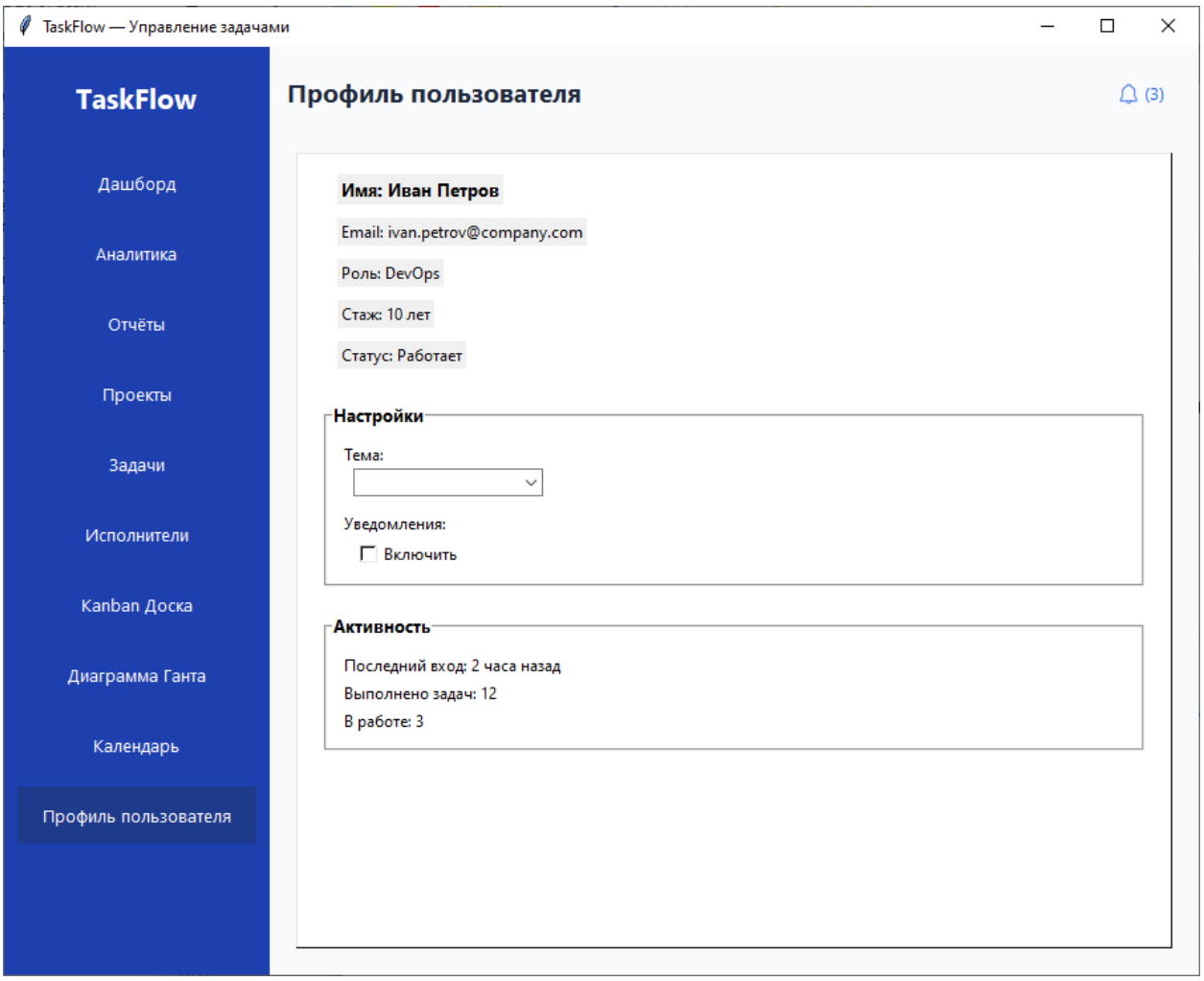


Рисунок 4.17 – Внешний вид вкладки «Профиль пользователя»

Если в настройка включены уведомления, то в пользователю будут поступать уведомления об основных действиях в системе, таких как, завершение задачи или проекта, назначение новой задачи на пользователя, появление задачи с истекшим сроком решения. Уведомления при поступлении появляются в правой нижней части экрана в виде всплывающего окна и пропадают при отсутствии действий см ними через 10 секунд. Вернуться к полученным уведомлениям можно нажав на пиктограмму колокольчика в правом верхнем углу программы. При выполнении данного действия появится дополнительное окно «Уведомления». Внешний вид окна представлен на Рисунке 4.18.

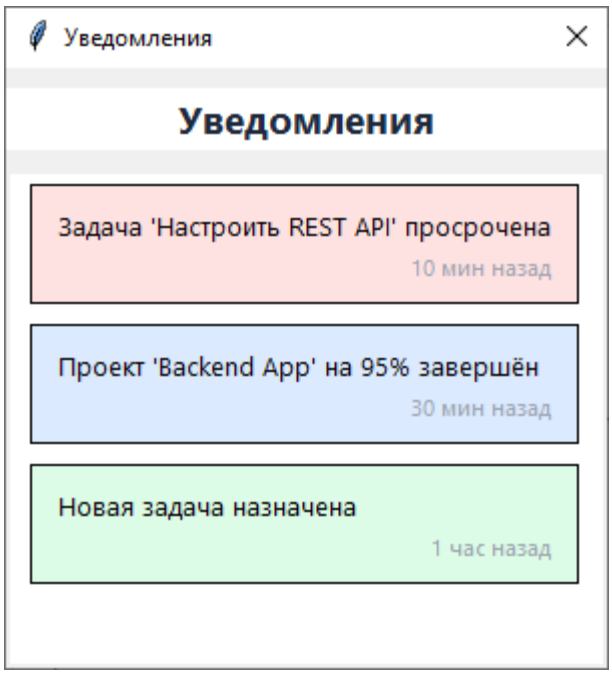


Рисунок 4.18 – Внешний вид окна «Уведомления»

Программа обладает потенциалом для дальнейшего расширения функционала. Планируемые доработки включают:

- управление пользователями в части назначения дополнительных ролей (администратор, руководитель проекта, тимлид);
- разграничение прав доступа для обеспечения безопасности данных;
- настройка шаблонов: создание шаблонов задач и проектов для ускорения процесса планирования и стандартизации рабочих процессов;
- экспорт отчетов в форматы PDF и Excel.

Для наглядного понимания взаимодействия пользователя с программой можно рассмотреть пример типового пользовательского пути:

1. Пользователь открывает программу и видит главную страницу «Дашборд», где представлен обзор текущих проектов, задач и ключевых показателей.

2. В разделе "Проекты" пользователь создает новый проект, добавляет описание, даты начала и окончания

3. В разделе "Задачи" пользователь формирует список задач для нового проекта, распределяет задачи на исполнителей, устанавливает приоритеты и настраивает связи между задачами (например, зависимости "финиш-старт").

4. В разделе "Исполнители" пользователь анализирует их эффективность по различным критериям и корректирует распределение задач при необходимости.

5. Для планирования работы пользователь использует интерактивный календарь загрузки..

6. Наконец, в разделе "Аналитика и отчеты" пользователь формирует сводные отчеты по проектам, задачам и исполнителям и предоставляет руководству для принятия решений.

Таким образом, программа предлагает удобный и интуитивно понятный интерфейс, позволяющий пользователям эффективно управлять проектами, задачами и ресурсами. Возможности доработки обеспечивают гибкость системы и её адаптацию под меняющиеся потребности бизнеса

4.4 Экспериментальное подтверждение эффективности разработанных моделей и алгоритмов

Для обоснования эффективности предложенных алгоритмов были проведены эксперименты, целью которых являлся сравнительный анализ результатов работы разработанных алгоритмов и результаты в случае произвольного распределения исполнителей по задачам без учета какой-либо коррекции.

Каждый эксперимент заключался в сравнении результатов, полученных без использования моделей и алгоритмов, предложенных в главе 2 и главе 3, и с использованием соответствующего математического и алгоритмического обеспечения. Кроме того, в проект вносились незначительные изменения. Она могли заключаться в изменении длительности задачи или недоступности какого-либо исполнителя в течение случайного интервала времени (имитация болезни/командировки).

Фрагменты результатов эксперимента представлены в Таблице 4.1.

Таблица 4.1 - Фрагменты результатов эксперимента

Номер проекта	Количество задач	Критический путь по средним значениям	Критический путь с параллельным решением задачи (10)	Критический путь при корректировке (болезнь\командировка сотрудника) - расчет без коррекции	Критический путь при корректировке (болезнь\командировка сотрудника) - расчет с коррекцией
1	20	62,00	59,72	74,40	61,51
2	16	24,63	23,18	29,55	26,19
3	14	57,14	53,88	68,57	60,89
4	22	72,13	68,96	86,56	71,02
5	17	49,67	47,88	59,60	49,32
6	16	31,83	30,14	38,20	34,06
7	12	53,15	49,59	63,78	56,04

Полученные результаты свидетельствуют об эффективности разработанных моделей и алгоритмов.

Элементы программного обеспечения зарегистрированы в ФИПС []. Полученная система внедрена в ЗАО «Микрон» (г. Липецк) в виде программного средства, обеспечивающего автоматизацию оценки общей длительности планируемых проектов и назначения исполнителей для каждой из отдельных задач. Эффект от внедрения заключается в увеличении точности оценки длительности проектов, повышение эффективности распределения загрузки исполнителей и уменьшении времени незапланированных простоев производства.

Для оценки эффективности работы комплекса моделей проведен сравнительный анализ деятельности до и после внедрения программных моделей. В результате эффективность управления повысилась в среднем на 5%.

Таким образом, разработано программное обеспечение системы управления деятельностью IT-компании как организационной системы,

учитывающее квалификацию специалистов, их занятость, тип задач и обеспечивающее наилучшую организацию деятельности IT-компании путем планирования времени начала каждой из задач проекта и подбора для нее исполнителя.

4.5 Выводы

1. Для управления IT-компанией как организационной системой требуется наличие механизмов, позволяющих организовать процесс работы исполнителей путем закрепления за каждым из них множества задач, определять график проекта и корректировать принятые решения при возникающих изменениях. Для этого необходимо программное средство, позволяющее автоматизировать данный процесс управления.

2. Разработанная структура программного обеспечения системы управления деятельностью IT-компании предусматривает возможность интеграции подсистем имитационного моделирования, планирования, оптимизации и базы данных, что позволяет автоматизировать процесса управления IT-компанией путем реализации разработанных моделей и алгоритмов.

3. Проведенные опыты, анализирующие величину критического пути с использованием разработанных моделей и алгоритмов и без них, демонстрируют эффективность предлагаемых решений.

4. Элементы программного обеспечения зарегистрированы в ФИПС. Полученная система внедрена в ЗАО «Микрон» (г. Липецк) в виде программного средства, обеспечивающего автоматизацию оценки общей длительности планируемых проектов и назначения исполнителей для каждой из отдельных задач.

ЗАКЛЮЧЕНИЕ

В процессе выполнения диссертационного исследования были получены следующие основные результаты:

1. Проведен анализ существующих подходов к управлению процессами проектной деятельности в IT-компании как организационной системы в рамках поиска наилучшего подхода к управлению коллективом исполнителей, распределению задач и планирования проектов.

2. Разработана модель управления проектом в IT-компании как организационной системе, обеспечивающая наилучшую организацию деятельности IT-компании путем планирования времени начала каждой из задач проекта и подбора для нее исполнителя.

3. Разработана математическая модель оценки сроков выполнения как отдельных задач, так и всего проекта данной организационной системы, обеспечивающая повышенную точность по сравнению с аналогами.

4. Разработаны алгоритмы организации работ в IT-компании, обеспечивающие возможность повышения качества управления деятельностью IT-компании за счет более рационального использования человеческих ресурсов и более точных оценок времени выполнения как всего проекта, так и его отдельных заданий.

5. Разработана структура программного обеспечения системы управления деятельностью IT-компании как организационной системы, обеспечивающая автоматизацию процесса управления IT-компанией путем реализации разработанных алгоритмов. Элементы программного обеспечения системы управления зарегистрированы в ФИПС № 2024663794.

6. Проведено экспериментальное исследование эффективности разработанных моделей и алгоритмов управления IT-компанией как организационной системой с точки зрения длительности IT-проекта. Результаты эксперимента демонстрируют эффективность предложенных решений в среднем на 4,91% в обычных условиях и на 17,29% в случае возникновения внештатных ситуаций.

Рекомендации и перспективы дальнейшей разработки темы

1. Результаты диссертационного исследования рекомендуются к применению в задачах управления в любых организационных системах, отличительной чертой которых является большая неопределенность длительности решения задач, разная квалификация и другие характеристики исполнителей, а также возможная коррекция задач из-за изменившихся требований заказчика и других факторов.

2. Дальнейшая разработка темы будет направлена на уточнение параметров оценок и критериальных коэффициентов, а также расширение системы поддержки принятия решений, необходимой для эффективного управления организационной системой, в которой будут использоваться результаты данного исследования.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Аксенов В.В. Метаэвристические методы решения задач комбинаторной оптимизации [Электронный ресурс]. Режим доступа: http://omega.sp.susu.ac.ru/books/conference/PaVT2009/papers/Posters/003.pdf/

2. Алгоритм управления временными параметрами управления проекта на основе системы контрольных точек. Вестник Воронежского государственного университета. Серия: Системный анализ и информационные технологии. 2022; 4: 39-51.

3. Анализ рисков инвестиционных проектов / И.В. Трегуб, А.В. Облакова // Вестник Финансовой академии. – 2023. – №2. – С. 23-33.

4. Арчибальд Р.Д. Управление высокотехнологичными программами и проектами. Москва: Компания АйТи; ДМК Пресс; 2004. 472 с.

5. АС Буткевич Современные подходы в управлении проектами, CyberLeninka, 2023.

6. Ахметов К. Практика управления проектами. М.: Русская редакция, 2004.

7. Ахо А., Хопкрофт Дж., Ульман Дж. Структуры данных и алгоритмы. М.: Вильямс, 2007. - 384 с.

8. Бадмаев, Е. З. Проектное управление в развитии предприятия, 2020

9. Барикаев Е.Н. Методы экспертных оценок / Е.Н. Барикаев, В.З. Черняк. — EDN RSCSIB // Вестник Московского университета МВД России. — 2013. — № 12. — С. 184-189.

10. Батищев Д.И. Генетические алгоритмы решения экстремальных задач: учебное пособие. Воронеж: ВГТУ, 1995. 65 с.

11. Батищев Д.И., Неймарк Е.А., Старостин Н.В. Применение генетических алгоритмов к решению задач дискретной оптимизации. Нижний Новгород, 2007, 85 с.

12. Беленький А. С. Исследование операций в транспортных системах: идеи и схемы методов оптимизации планирования. Москва: Мир, 1992. 582 с.

13. Беляева М.А., Буреш О.В., Шаталова Т.Н. Разработка интегрированной системы поддержки принятия решений по управлению проектами в условиях неопределенности // Вестник Оренбургского государственного университета. 2011. № 13. С. 43-48.

14. Бойкова М. В., Колобова И. Н., Кузнецов С. С. Управление проектами: Учебник. М.: Российская таможенная академия, 2018.

15. Болнокина Е. В., Олейникова С. А. "Определение оптимального состава исполнителей для многостадийной обслуживающей системы". — Журнал "Математические основы информационных технологий", 2019.

16. Бондаренко Ю.В. Выбор методов оценки при формировании кадрового состава проектных команд / Ю.В. Бондаренко, И.С. Никитин, Н.Ю. Калинина, А.М. Ходунов // Вестник Южно-Уральского государственного университета. Серия: Компьютерные технологии, управление, радиоэлектроника. 2020. Т. 20, № 2. С. 116-124. DOI: 10.14529/ctcr200211

17. Бурков В.Н. Новиков Д.А. Как управлять организациями. - М.: СИНТЕГ, 2004. - 188 с.

18. Бурков В.Н., Новиков Д.А. Как управлять проектами. – М.: Синтег, 1997. – 188 с.

19. Бусленко В.Н. Автоматизация имитационного моделирования сложных систем.-М.: Наука, 1977.- 240 с.

20. Буткевич, А.С. Анализ эффективности управления ИТ-проектами / А. С. Буткевич // Инновации и инвестиции. 2022. № 11. С. 86-90.

21. Вагнер Г. Основы исследования операций. М.: Мир, 1972. Т. 1. 336 с.

22. Винокуров А.С., Николаев С.В., Баженов Р.И. Реализация метода PERT в программной системе GanntProject // Nauka-Rastudent.ru.-2015.-№ 6(18). - С. 22.

23. Вольфсон Б. Гибкое управление проектами и продуктами / Б. Вольфсон. – СПб.: Питер, 2020. – 144 с.

24. Глушков А.Ю. Математическая модель эффективного управления проектами путем оптимального распределения ресурсов. Системы управления и информационные технологии. 2020;(1):75-78.

25. Голенко-Гинзбург Д.И. Стохастические сетевые модели планирования и управления разработками / Д.И. Голенко-Гинзбург: Монография. - Воронеж: "Научная книга", 2010. - 284 с.

26. ГОСТ 53892-2010: Руководство по оценке компетентности менеджеров проектов

27. Дейт К. Дж. SQL и реляционная теория. Как грамотно писать код на SQL. М.: Символ - плюс, 2010. - 474 с.

28. Дятчина А.В. Программное обеспечение для автоматизации процесса управления деятельностью IT-компании // Управление программным инжинирингом. Труды открытой Международной научно-практической конференции. Воронеж, 2025. С. 49-53.

29. Дятчина А.В., Олейникова С.А. Метод решения задачи планирования работ в IT-проекте и назначения им специалистов // Моделирование, оптимизация и информационные технологии. 2024. Т. 12, № 4. DOI: 10.26102/2310-6018/2024.47.4.009.

30. Дятчина А.В., Олейникова С.А. Моделирование выполнения стохастического проекта в условиях коррекции решения его отдельных задач // Актуальные проблемы прикладной математики и механики. Сборник трудов Международной научной конференции. Воронеж, 2025. С. 512-518.

31. Дятчина А.В., Олейникова С.А. Оптимизация распределения исполнителей по работам в многостадийных стохастических системах. Свидетельство о регистрации программы для ЭВМ RU 2024663794, 11.06.2024. Заявка от 04.06.2024.

32. Дятчина А.В., Олейникова С.А. Оценка стохастических характеристик отдельных работ в IT-проектах // Научная опора Воронежской

области. Сборник трудов победителей конкурса научно-исследовательских работ студентов и аспирантов ВГТУ по приоритетным направлениям развития науки и технологий. Воронеж, 2024. С. 188-191.

33. Дятчина А.В., Олейникова С.А. Оценка стохастических характеристик отдельных работ в IT-проектах. В сборнике: Научная опора Воронежской области: Сборник трудов победителей конкурса научно-исследовательских работ студентов и аспирантов ВГТУ по приоритетным направлениям развития науки и технологий, 15-19 апреля 2024 года, Воронеж, Россия. Воронеж: Воронежский государственный технический университет; 2024. С. 188-191.

34. Дятчина А.В., Олейникова С.А. Формализация многокритериальной задачи о назначениях с временными ограничениями // Прикладная математика и информатика: современные исследования в области естественных и технических наук. Сборник материалов IX Международной научно-практической конференции (школы-семинара) молодых ученых. Тольятти, 2023. С. 119-123.

35. Дятчина А.В., Олейникова С.А., Акинин А.А. Модель управления IT-проектом с возможной последующей коррекцией отдельных задач // Системы управления и информационные технологии. 2024. № 4 (98). С. 85-88.

36. Дятчина А.В., Олейникова С.А., Акинин А.А. Модель управления IT-проектом с возможной последующей коррекцией отдельных задач // Системы управления и информационные технологии.

37. Дятчина А.В., Олейникова С.А., Недикова Т.Н. Разработка проблемно-ориентированной системы для решения оптимизационной задачи управления потоком поступающих заявок // Вестник Воронежского государственного технического университета. 2023. Т. 19. № 4. С. 37-43.

38. Дятчина А.В., Олейникова С.А., Селищев И.А. Имитационная модель для исследования длительности выполнения работы в случае ее возможного повторного выполнения // Математическое моделирование

процессов и систем. Материалы XIII Международной молодежной научно-практической конференции. Отв. редактор С.В. Викторов. Стерлитамак, 2023. С. 135-141.

39. Иванов С. А. Современные методологии управления ИТ-проектами: Waterfall, Agile и гибридная модель // Научный Лидер. 2023. №48 (146). С. 12-16.

40. Катаева В.И., Козырев М.С. Методы принятия управленческих решений. М. - Берлин: Директ-Медиа, 2015. 196 с.

41. Кендалл И. Современные методы управления портфелями проектов и офис управления проектами: Максимизация ROI / И. Кендалл, К. Роллинз; пер. с англ. - М.: ЗАО "ПМСОФТ", 2004. - 576 с.

42. Кобзарь А. И. Прикладная математическая статистика. Для инженеров и научных работников. - М.: Физматлит, 2006. - 816 с.

43. Козлов А.С. Проектирование и исследование бизнес-процессов: [Текст]: учебное пособие / А.С. Козлов. М.: Флинта: МПС, 2006. 272 с.

44. Кон М. Agile: Оценка и планирование проектов. Москва: Альпина Паблишер; 2022. 418 с.

45. Короходкина Ю. И., Гагарина С. Н. Современные методы управления проектами/ Journal of Economy and Business, vol. 1-2 (83), 2022, - С. 38-43 - [Электронный ресурс] - 2022 - Режим доступа: https://cyberleninka.ru/article/n/sovremennye-metody-upravleniya-proektami-1

46. Косарева, И.Н., Самарина, В.П. Особенности управления предприятием в условиях цифровизации //Вестник Евразийской науки, 2019, №3, Т.11, С.1-9

47. Коул Р., Скотчер Э**.** Блистательный Agile. Гибкое управление проектами с помощью Agile, Scrum и Kanban. СПб.: Питер, 2019.

48. Кофман А., Дебазей Г. Сетевые методы планирования: Применение системы ПЕРТ и ее разновидностей при управлении производственными и научно-исследовательскими проектами. Москва: Издательство "Прогресс"; 1968. 181 с.

49. Кочетов Ю.А. Вероятностные методы локального поиска для задач дискретной оптимизации. В сборнике: Дискретная математика и ее приложения: Сборник лекций молодежных и научных школ по дискретной математике и ее приложениям. Москва: Издательство центра прикладных исследований при механико-математическом факультете МГУ; 2001. С. 84-117.

50. Курейчик В.В., Полупанова Е.Е. Эволюционная оптимизация на основе алгоритма колонии пчел // Известия ЮФУ. Технические науки. - 2009. - № 12 (101). - С. 4146.

51. Лелякова Л. В., Харитонова А. Г., Чернышева Г. Д. Прикладные задачи о назначениях (модели, алгоритмы решения) // Вестник Воронежского государственного университета. Серия: Системный анализ и информационные технологии. 2017. № 2. С. 22-27.

52. Лелякова Л. В., Харитонова А.Г., Чернышева Г.Д. Прикладные задачи о назначениях (модели, алгоритмы решения)// Вестник Воронежского государственного университета. - 2017. - № 2. - С. 22-27.

53. Луценко Е.В., Коржаков В.Е.Системно-когнитивный подход к решению обобщенной задачи о назначениях // Вестник адыгейского государственного университета. Серия 4: Естественно-математические и технические науки. 2010. № 2. С.98-106.

54. Макарова В.А. Анализ и оценка экономической эффективности риск-менеджмента // Стратегические решения и риск-менеджмент. 2015. № 3. С. 72—83.

55. Макдональд Н. Успешное управление проектами. - М.: Дело, 2020.

56. Математические основы управления проектами: учеб. пособие / С. А. Баркалов, В. И. Воропаев, Г. И. Секлетова [и др.]; под ред. В. Н. Буркова. - М.: Высш. шк., 2005. - 423 с.

57. Мельников Б.Ф. О классической версии метода ветвей и границ / Б.Ф. Мельников, Е.А. Мельникова // Компьютерные инструменты в образовании. 2021. № 1. С. 21-44. DOI: 10.32603/2071-2340-2021-1-21-45

58. Мери Л., Вилитинский В. Управление проектами. - СПб.: Питер, 2019;

59. Методология IDEF0. Стандарт (русская версия). М.: Мета-Технология, 2020. 117 с.

60. Моров В.А. Применение генетического алгоритма к задачам оптимизации. Реализация генетического алгоритма для задачи коммивояжера // Вестник АмГУ, 2012.

61. Морозов, А.В. Методы оценки эффективности инновационных проектов // Научно-практические исследования. — 2020. — N° 6-1 (29). — С. 51-54.

62. Новиков Д. А. Управление проектами: организационные механизмы / Д. А. Новиков. -М.: ПМСОФТ, 2007. - 140 с

63. Новиков, Ф.А. Дискретная математика для программистов: учебное пособие для вузов. - СПб.: Питер, 2003. - 304 с.

64. Олейникова С. А. Вычислительный эксперимент для анализа закона распределения случайной величины, описывающей длительность проекта в задачах сетевого планирования и управления / С. А. Олейникова // Экономика и менеджмент систем управления. - 2013. -№ (9). - С. 91-97

65. Олейникова С.А. Математические модели и методы оптимизации функционирования сложных обслуживающих систем со стохастическими параметрами: дис… докт. техн. наук. – Воронеж, 2016. – 364 с

66. Олейникова С.А., Дятчина А.В. Алгоритм решения задачи организации работ в it-компании // Интеллектуальные информационные системы. Воронеж, 2025. С.84-88.

67. Олейникова С.А., Дятчина А.В. Анализ влияния коррекции выполнения задач и их исполнителей на временные характеристики длительности стохастического проекта // Оптимизация и моделирование в

автоматизированных системах. Труды Международной молодежной научной школы. Воронеж, 2025. С. 79-84.

68. Олейникова С.А., Дятчина А.В. Анализ влияния коррекции выполнения задач и их исполнителей для временные характеристики длительности стохастического проекта// Оптимизация и моделирование в автоматизированных системах: тр. Междунар. молодежной научной школы. - Воронеж: Воронежский государственный технический университет, 2025. с.86-90.

69. Олейникова С.А., Дятчина А.В. Моделирование и анализ длительности стохастического проекта// Вестник Пензенского государственного университета. 2025. № 1 (49). С. 98-101.

70. Олейникова С.А., Дятчина А.В. Разработка специального математического обеспечения системы управления потоком поступающих заявок // Вестник Воронежского государственного технического университета. 2023. Т. 19. № 2. С. 33-37.

71. Олейникова С.А., Дятчина А.В., Аралов М.Н. Математическая модель оценки сроков IT-проекта и его отдельных задач// Системы управления и информационные технологии. 2025. №2(100). С. 4-7.

72. Олейникова С.А., Дятчина А.В., Политов В.А. Структура программного обеспечения для управления деятельностью IT-компании. Моделирование, оптимизация и информационные технологии. 2025;13(1). URL: https://moitvivt.ru/ru/journal/pdf?id=1843 DOI: 10.26102/2310-6018/2025.48.1.044.

73. Олейникова С.А., Дячина А.В. Алгоритм коррекции план-графика выполнения задач в IT-проекте // Информационные технологии в моделировании и управлении: подходы, методы, решения. Сборник трудов Тольятти, 2025.

74. Олейникова С.А., Дячина А.В. Сравнительный анализ эволюционных алгоритмов решения задачи планирования IT-проектов // Проектное управление в строительстве, 2024. № 2 (31). С. 100-106.

75. Олейникова С.А., Менкова Е.С. Динамическая задача о назначении единичного задания с временными ограничениями // Вестник Воронежского государственного технического университета. 2020. Т. 16. № 6. С. 19-24.

76. Олейникова, С. А. Математическая модель и оптимизационная задача составления расписания для мультипроектной системы с временными и ресурсными ограничениями и критерием равномерной загрузки / С. А. Олейникова // Вестник Воронежского государственного технического университета.-2013.-Т9.-№ 6-3.-С. 58-61.

77. Основы управления проектами / А. В. Аверин, В. В. Жидиков, И. В. Корнева, 2020

78. Пападимитриу Х., Стайглиц К. Комбинаторная оптимизация. Алгоритмы и сложность. М.: Мир, 1984. 512 с.

79. Рубин К.С. Основы Scrum. Практическое руководство по гибкой разработке ПО. - М.: Диалектика-Вильямс, 2019. - 544 с.

80. Сазерленд Дж. Scrum. Революционный метод управления проектами. Москва: Манн, Иванов и Фербер; 2015. 288 с.

81. Салахов А.Ж., Байрашев И.Д. Оценка рисков инвестиционных проектов // Экономика и социум. 2016. № 3. С. 1093-1100.

82. Серебрякова Г.В., Незамайкин И.В. Ценностный контур развития теории управления // Экономика и управление: анализ тенденций и перспектив развития. - 2013. - №6. - С.12-20.

83. Серышев Р.В. Оптимизация бизнес-процессов и проектирование бизнес-систем (Тезисы) // Реформы в России и проблемы управления — 2010 [Текст]: материалы 25-й Всероссийской научной конференции молодых ученых и студентов. Вып. 1; Государственный университет управления. М.: ГУУ, 2010. 264 с.

84. Серышев Р.В. Эволюция процессного подхода к управлению (тезисы) // Актуальные проблемы управления — 2011 [Текст]: Материалы 16-

й Всероссийской научно-практической конференции. Вып. 3; Государственный университет управления. М.: ГУУ, 2011. 302 с.

85. Стеллман Э., Дженнифер Г. / Постигая Agile: ценности, принципы, методологии / Пер. с англ. С. Пасерба. - 3-е изд. - М.: Манн, Иванов и Фербер, 2019. - 441 с.

86. Таха Х. А. Введение в исследование операций. 7-е изд. Москва [и др.]: Вильямс, 2005. 901 с.

87. Терентьева З. С. Гибкие методы управления проектами, анализ и сравнение / З. С. Терентьева, И. А. Хализова // Азимут научных исследований: экономика и управление. -2019. - Т. 8, № 1 (26). - С. 374-376.

88. Ткаченко И.Н., Сивокоз К.К. Использование гибких технологий Agile и Scrum для управления стейкхолдерами проектов // Управленец. - 2017. - №4 (68). - С. 85-95.

89. Управление проектами в компании: определение и решение ключевых проблем. [Электронный ресурс]. – URL: https://www.advanta-group.ru/blog/upravlenie-it-proektami/

90. Управление проектами в России и мире: современные тенденции» – С. В. Мастеров, [Электронный ресурс] – 2023 - Режим доступа: https://cyberleninka.ru/article/n/upravlenie-proektami-v-rossii-i-mire-sovremennye-tendentsii

91. Царькова Е.В. Методы управления проектами в условиях информационной неопределенности / Е.В. Царькова// Правовая информатика. — 2019. — № 4. — С. 29-39.

92. Цителадзе Д. Д. Алгоритмизация процессов управления проектом / Д. Цителадзе // Научные исследования и разработки. Российский журнал управления проектами. - 2020. -Т. 9, № 1. - С. 12-21.

93. Шайдурова А. Особенности управления ИТ-проектами // Научный электронный архив. URL: http://econf.rae.ru/article/11650

94. Штовба С.Д. Муравьиные алгоритмы // Exponenta Pro. Математика в приложениях. - 2003. - №4. - С. 70-75.

95. Щукина Н.А. Некоторые подходы к решению задачи о назначениях // Проблемы экономики и менеджмента. 2016. № 5(57). С. 169 - 174.

96. Э. Ларсон, К. Грей. Управление проектами: Практическое руководство/. Пер. с англ. - М.: Издательство «Дело и Сервис»; 2013. 784 с.

97. Ядута А. З., Яцынюк Д. А., Лескин К. А. О методах свертывания критериев в многокритериальной задаче о назначениях // Высокие технологии и инновации в науке: Материалы междунар. науч. конф.: сборник избранных статей / отв. за вып. Л. А. Павлов. Санкт-Петербург, 2019. С. 232-237.

98. A Guide to the Project Management Body of Knowledge (PMBOK® Guide) — Seventh Edition and The Standard for Project Management. Pennsylvania: Project Management Institute; 2021. 250 p.

99. Bolnokina E.V. Formalizatsiya zadachi vybora mekhanizmov stimulirovaniya v zadache organizatsionnogo upravleniya mnogostadiynoy proizvodstvennoy sistemoy/ E.V. Bolnokina, S.A. Oleynikova // Sistemy upravleniya i informatsionnye tekhnologii, No. 4(74), 2018. – pp. 26-29.

100. Borshchev A. The Big Book of Simulation Modeling. Multimethod Modeling with AnyLogic 6 [Электронный ресурс] // AnyLogic.ru. URL https://www.anylogic.ru/resources/books/big-book-of-simulation-modeling/.

101. Cohen R., Katzir L., Raz D. An Efficient Approximation for the Generalized Assignment Problem // Information Processing Letters. 2006. Vol. 100, № 4. P. 162-166.

102. Comparison of Machine Learning Algorithms for Software Project Time Prediction / Han W. J., Jiang L. X., Lu T. B., Zhang X. Y. // International Journal of Multimedia and Ubiquitous Engineering. 2015. Vol. 10, Issue 9. P. 1-8.

103. D. B. Shmoys and Eva Tardos. An approximation algorithm for the generalized assignment problem. Mathematical Programming, 62(3) 1993. – pp. 461-474.

104. Developing a Mathematical Model for Scheduling and Determining Success Probability of Research Projects Considering Complex-Fuzzy Networks / Norouzi G., Heydari M., Noori S., Bagherpour M. // Journal of Applied Mathematics. 2015. Vol. 2015. P. 1-15.

105. Dyatchina A.V. Formalizing the task of managing the flow of incoming tasks with time constraints // Сборник научных статей XVIII международной научно-практической конференции «Антропоцентрические науки в образовании». Воронеж, 2023. С. 280-283.

106. Harold W. Kuhn The Hungarian Method for the assignment problem // Naval Research Logistics Quarterly. 1955. No 2. P. 83-97.

107. Haughey D. A brief history of project management // Projectsmart [Электр. ресурс]: http://www.projectsmart.co.uk/brief-history-of-project-management.html.

108. Land A.H. and Doig A.G. An automatic method of solving discrete programming problems // Econometrica. 1960. V. 28. P. 497-520.

109. Larson, E. and Gray, C., 2020. ISE Project Management: The Managerial Process. 8th ed. McGraw-Hill.

110. Little J.D.C., Murty K.G., Sweeney D.W., and Karel C. An Algorithm for the Traveling Salesman Problem // Operations Research. 1963. No 11. P. 972-989.

111. Martinelli R.J., Milosevic D.Z. Project Management ToolBox: Tools and Techniques for the Practicing Project Manager. New Jersey: John Wiley -amp; Sons, Inc.; 2016. 480 p.

112. P2M – Управление проектами и программами на основе системы знаний P2M: Ярошенко Ф. А., Бушуев С. Д., Танака Х., Киев: 2011. Также: Pod ред. Ярошенко Ф. А., К.: Новый друк, 2010.

113. Perry C., Greig I. Estimating the mean and variance of subjective distributions in PERT and decision analysis. Management. Science. 1975; 21(12). pp. 1477-1480.

114. Project Management Institute PMBOK® Guide – Seventh Edition and The Standard for Project Management (Russian), 2021

115. Saebi, T., Foss, N.J., Linder, S. Social Entrepreneurship Research: Past Achievements and Future Promises. Journal of Management, 2019, Vol. 45, no. 1, pp. 70-95. DOI: 10.1177/0149206318793196.

116. Shitova M.V., Ovanesyan S.S. Structure of the Decision Support System in the Development of Software Products. System Analysis & Mathematical Modeling, 2021, vol. 3, no. 4, pp. 258-269.

117. Takha Kh. A. Vvedenie v issledovanie operatsiy, 7 izdanie.: Per. s angl. – M.: «Vil'yams», 2005. – 912 p.

118. Trietsch D., Baker K. R. PERT 21: Fitting PERT/CPM for use in the 21st century // International Journal of Project Management. 2012. Vol. 30, Issue 4. P. 490-502.

119. Vatutin E.I., Titov V.S., Emel'yanov S.G. Osnovy diskretnoy kombinatornoy optimizatsii. M.: Argamak- Media, 2016. – 270 p.

120. Vries, K. de. Case study methodology, in: Aranda K. (ed.) Critical Qualitative Health Research: Exploring Philosophies, Politics and Practices, Routledge, 2020, pp. 4152.

ПРИЛОЖЕНИЕ А – Акты о внедрении

УТВЕРЖДАЮ
Заместитель
генерального директора
ЗАО «Микрон»
«04» апреля 2025 г.

АКТ О ВНЕДРЕНИИ

Результаты научно-исследовательской работы Дягиной Анастасии Владимировны, связанные с разработкой моделей и методов управления деятельностью IT-компании, включая определение оптимальных сроков выполнения проекта, а также назначения им исполнителей, внедрены в деятельность компании ЗАО «Микрон» в виде программного средства, обеспечивающего автоматизацию оценки общей длительности планируемых проектов и назначения исполнителей для каждой из отдельных задач.

Эффект от внедрения заключается в снижении числа проектов, завершенных позже нормативного срока, за счет использования более точных оценок времени их выполнения и более рационального использования квалифицированных кадров.

Технический директор С.В. Рахимова

УТВЕРЖДАЮ

Проректор по учебной работе

ФГБОУ ВО «ВГТУ»

_____ А.И. Колосов

«29» августа 2025 г.

А К Т

внедрения результатов кандидатской диссертации в учебный процесс ФГБОУ ВО «Воронежский государственный технический университет»

Тема диссертации: «Управление процессами проектной деятельности IT-компании на основе оптимизационных моделей распределения исполнителей и временных ресурсов».

Автор: Дятчина Анастасия Владимировна

Научный руководитель: Олейникова Светлана Александровна

Выполненной в ФГБОУ ВО «Воронежский государственный технический университет» на кафедре автоматизированных и вычислительных систем в рамках основного научного направления «Информатика и вычислительная техника»

В период с «12» марта 2025 г. по н.в. внедрены в учебный процесс кафедры по группе научных специальностей 2.3 «Информационные технологии и телекоммуникации», научной специальности: 2.3.4 «Управление в организационных системах», на основании решения кафедры АВС от « 29 » августа 2025 г., протокол № 1 .

1. Вид результатов внедренных в учебный процесс: модели и алгоритмы планирования проектов и назначения отдельным работам исполнителей, разработанные в ходе диссертационного исследования.

2. Область применения: практические и лекционные занятия по дисциплине «Проектная деятельность», выполнение курсовых проектов, выпускных квалификационных работ.

3. Форма внедрения: разработанные в диссертационном исследовании схемы и механизмы были внедрены в образовательный процесс в виде моделей и алгоритмов управления проектами с целью обучения студентов

особенностям проектной деятельности, изучения подходов к коррекции графиков и особенностям управления коллективом исполнителей.

4. Эффект от внедрения. Повышение качества образования: применение новых моделей и алгоритмов планирования проектов и оценки их длительности в случае большого числа стохастических составляющих, а также разной квалификации исполнителей.

Научный руководитель диссертанта

_____ Олейникова С.А.
(подпись, Ф.И.О.)

«29» августа 2025 г.

Начальник УМУ

_____ Скляров К.А.
(подпись, Ф.И.О.)

«29» августа 2025 г.

Диссертант

_____ Дятчина А.В.
(подпись, Ф.И.О.)

«29» августа 2025 г.

Декан ФИТКБ

_____ Бредихин А.В.
(подпись, Ф.И.О.)

«29» 08 2025 г.

И.о. заведующего кафедрой АВС

_____ Барабанов В.Ф.
(подпись, Ф.И.О.)

«29» августа 2025 г.

ПРИЛОЖЕНИЕ Б – Свидетельство о государственной регистрации программы для ЭВМ

РОССИЙСКАЯ ФЕДЕРАЦИЯ

СВИДЕТЕЛЬСТВО

о государственной регистрации программы для ЭВМ

№ 2024663794

«Оптимизация распределения исполнителей по работам в многостадийных стохастических системах»

Правообладатели: *Дятчина Анастасия Владимировна (RU), Олейникова Светлана Александровна (RU)*

Авторы: *Дятчина Анастасия Владимировна (RU), Олейникова Светлана Александровна (RU)*

Заявка № **2024662812**

Дата поступления **04 июня 2024 г.**

Дата государственной регистрации в Реестре программ для ЭВМ *11 июня 2024 г.*

Руководитель Федеральной службы по интеллектуальной собственности

Ю.С. Зубов

ПРИЛОЖЕНИЕ В – Результаты имитационного моделирования

Ниже приведены результаты экспериментов с имитационной моделью исследовалась динамика выполнения проектов с различными вариантами сетевых структур (последовательно-параллельные и последовательные схемы) и распределениями времени выполнения работ. Цель моделирования заключалась в оценке критического пути проекта при варьировании параметров длительности задач и количестве прогонов модели (1001, 5001, 10001), а также в проверке чувствительности модели к смещению наиболее вероятных значений в сторону минимальных или максимальных оценок.

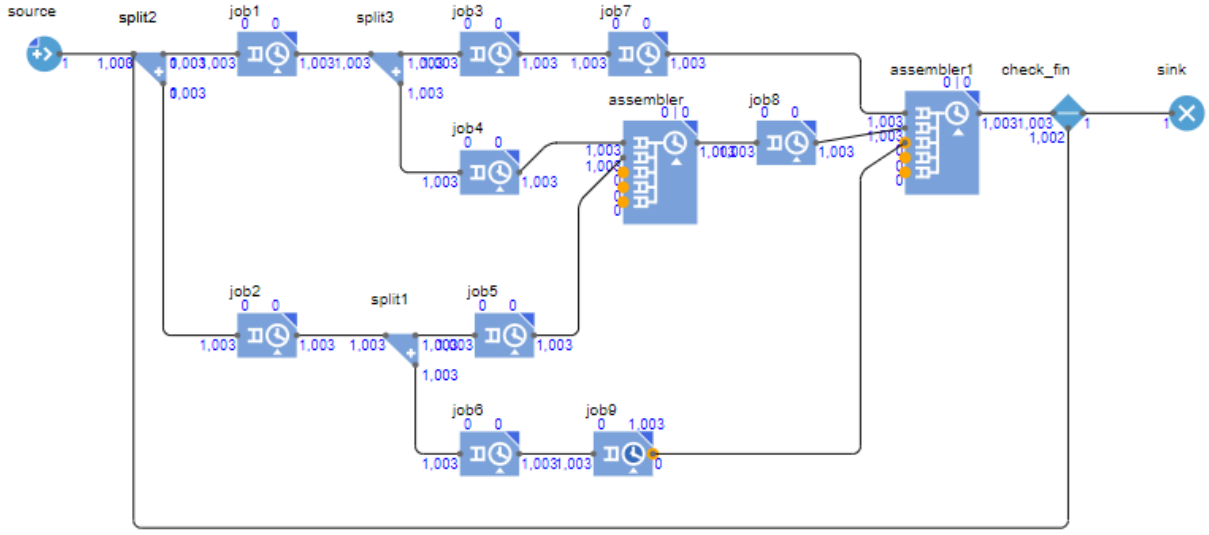
Ключевые параметры включали: количество задач (5 или 9), тип связности (параллельно-последовательная либо последовательная), распределение длительностей (a, m, b), а также число повторений. Условия экспериментов различались по характеру распределения временных параметров и числу итераций Монте-Карло.

Методы анализа основывались на сопоставлении результатов моделирования с ручными расчётами критического пути. Графики позволили оценить смещения распределений влево или вправо относительно среднего, а таблицы отразили различия по минимальным, максимальным и средним значениям.

Основные результаты показали, что при больших объёмах прогонов распределения стабилизируются, однако точное совпадение с ручными вычислениями достигается лишь по крайним (максимальным) значениям длительности. Для средних и минимальных значений наблюдаются устойчивые расхождения, отражающие ограниченность исходных предположений о форме распределений. В ряде случаев (эксперименты с 5 задачами) имитационная модель демонстрировала значительные отклонения от аналитических расчётов, что указывает на необходимость дополнительной валидации и уточнения параметров распределений. В целом анализ

подтвердил высокую чувствительность критического пути к изменению вероятностных оценок длительности работ и структуре сетевой модели.

<u>Последовательно-параллельный вариант, 9 работ.</u>



Распределения времени выполнения задач:

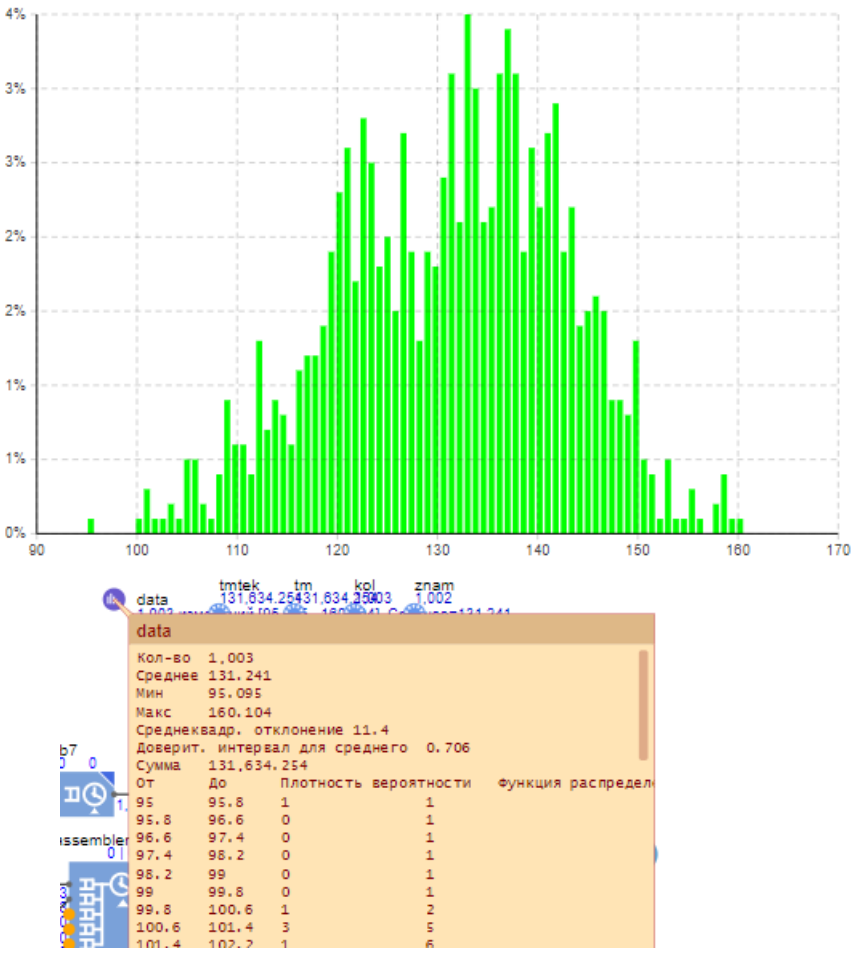
время в часах	Задача 1	Задача 2	Задача 3	Задача 4	Задача 5	Задача 6	Задача 7	Задача 8	Задача 9
A	5	10	12	2	36	10	1	22	14
B	12	22	17	8	91	36	4	58	33
M	7	15	13	4	52	21	3	37	24

Критический путь:

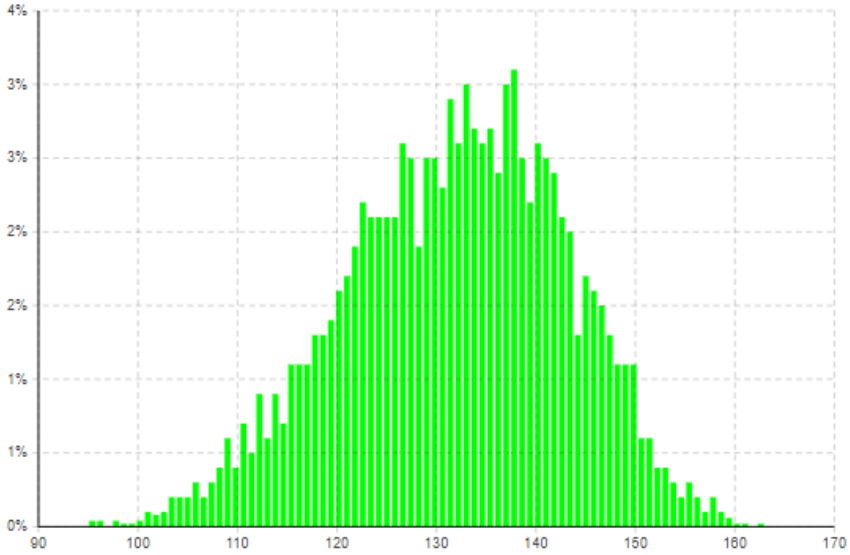
Критический путь проекта	время в часах
68	A
169	B
104	M

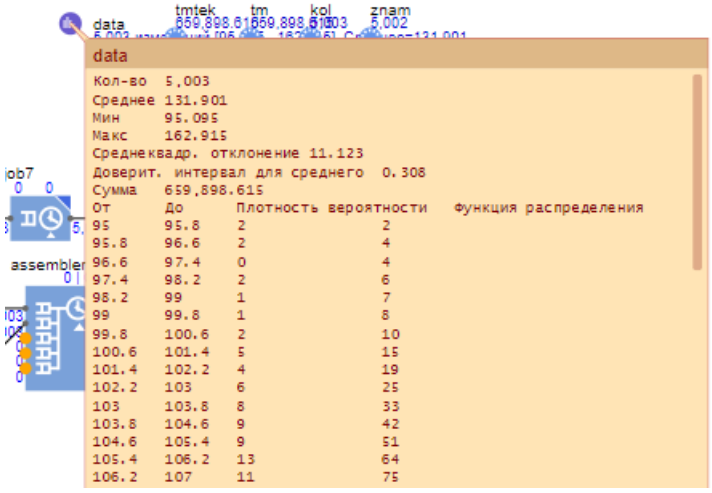
Вид – параллельно-последовательный
Кол-во задач – 9
Вероятность выполнения работ с 1 раза – 100 %

Эксперимент 1 - 1001.
Кол-во повторений – 1001

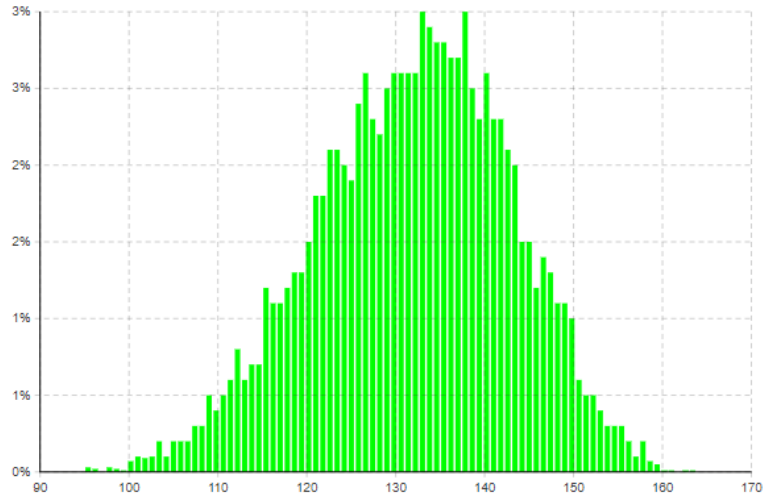


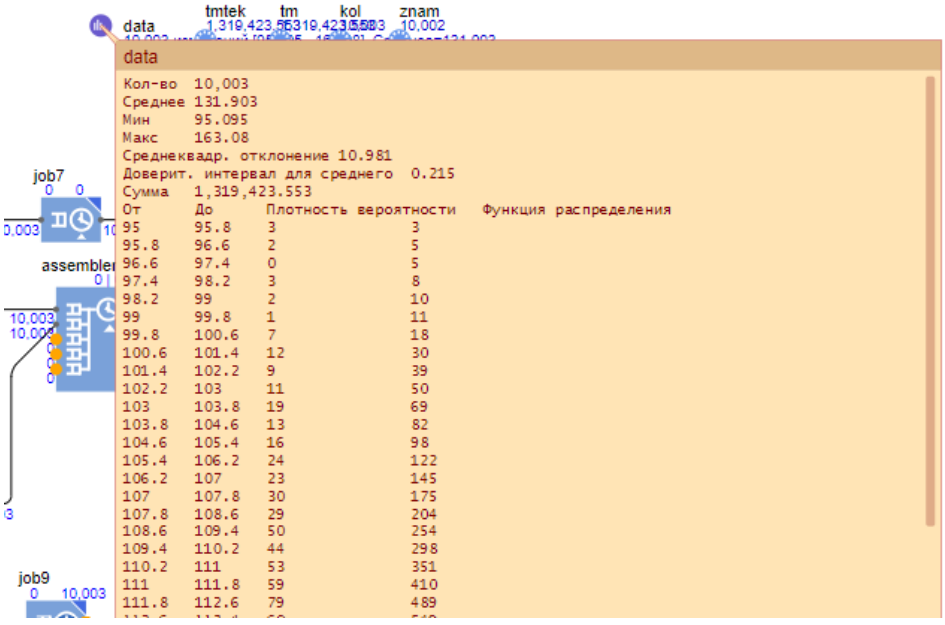
Эксперимент 2 – базовый прогон, 5001.
Кол-во повторений – 5001





Эксперимент 3 – базовый прогон, 10001.
Кол-во повторений – 10001





Модель показала устойчивые значения критического пути при увеличении числа повторов. Совпадение с ручными расчётами наблюдается лишь для максимальных значений, тогда как средние и минимальные стабильно завышены. Это отражает ограниченность исходного распределения и необходимость уточнения параметров.

эксперименты 1-3				
	1001	5001	10001	ручной
мин	95	95	95	68
макс	160	162	163	169
м	131	131	131	104

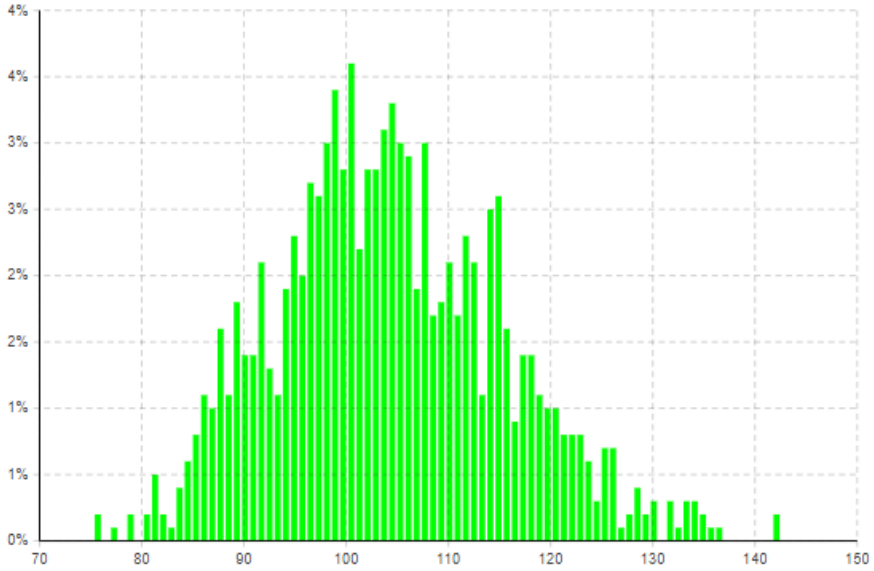
Эксперимент 7 - наиболее вероятное время выполнения работ сдвинуто в сторону максимального времени выполнения работ, 1001.
Кол-во повторений – 1001

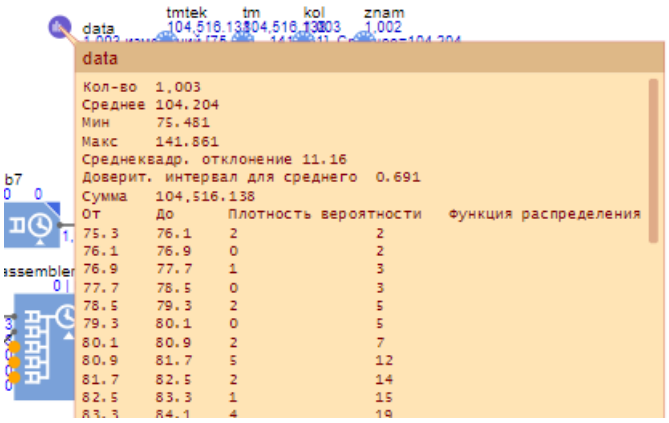
Распределения времени выполнения задач:

время в часах	Задача 1	Задача 2	Задача 3	Задача 4	Задача 5	Задача 6	Задача 7	Задача 8	Задача 9
A	5	10	12	2	36	10	1	22	14
B	12	22	17	8	91	36	4	58	33
M	10	19	16	7	82	31	3	45	29

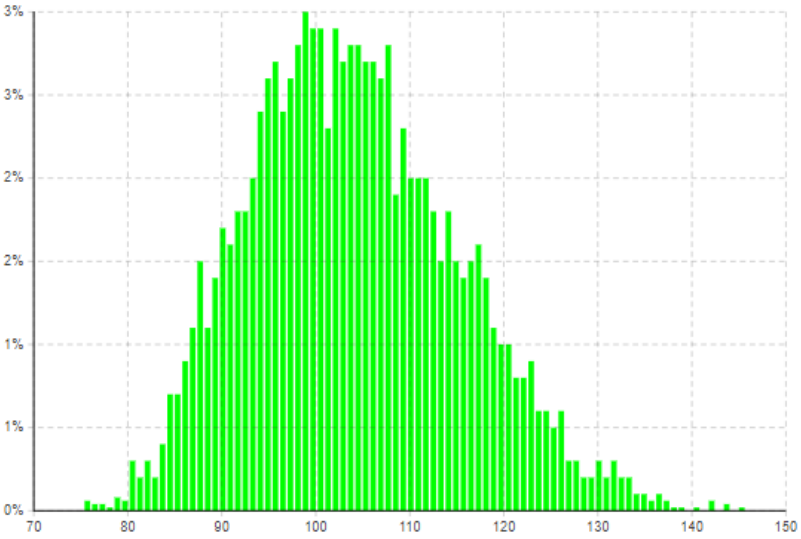
Критический путь:

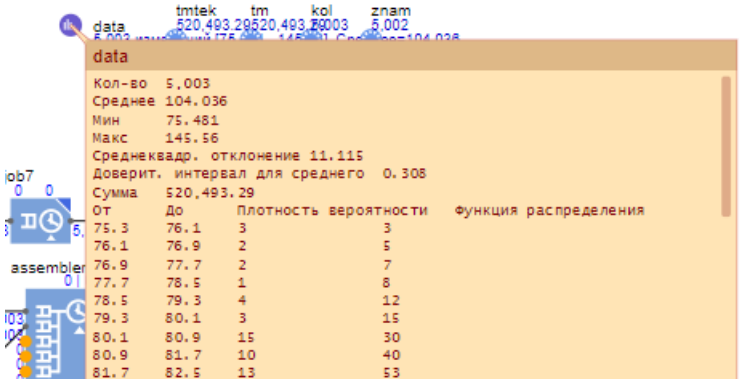
Критический путь проекта	время в часах
68	A
169	B
145	M



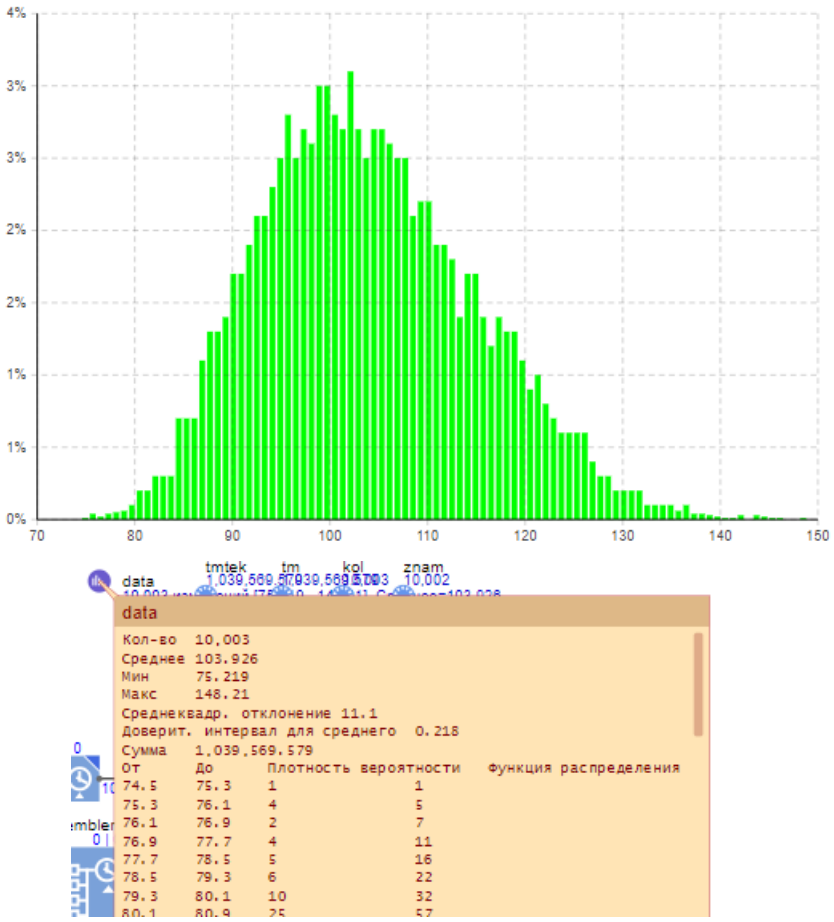


Эксперимент 8 - наиболее вероятное время выполнения работ сдвинуто в сторону максимального времени выполнения работ, 5001.
Кол-во повторений – 5001





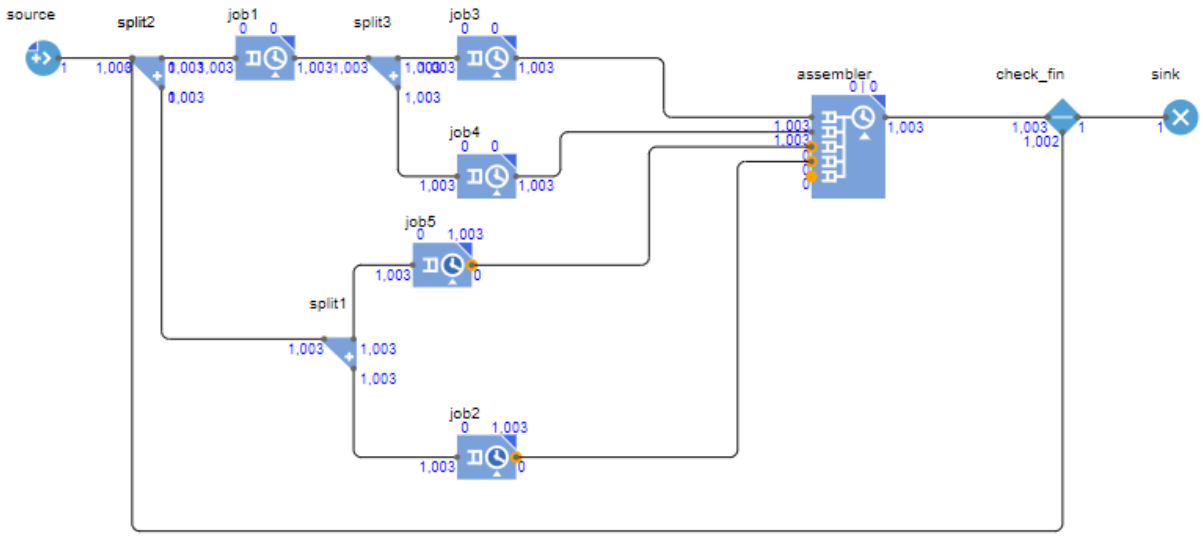
Эксперимент 9 - наиболее вероятное время выполнения работ сдвинуто в сторону максимального времени выполнения работ, 10001.
Кол-во повторений – 10001



Результаты фиксируют смещение распределений влево, что приближает минимальные значения к ручным подсчётам. При этом совпадение по максимальным значениям ухудшается, а средние остаются далеки от аналитики. Таким образом, смещение m усиливает чувствительность модели, но снижает точность в крайних сценариях.

эксперименты 7-9				
	1001	5001	10001	ручной
мин	75	75	75	68
макс	141	145	148	169
м	104	104	103	145

<u>Последовательно-параллельный вариант, 5 работ</u>



Вид – параллельно-последовательный
Кол-во задач – 5
Вероятность выполнения работ с 1 раза – 100 %

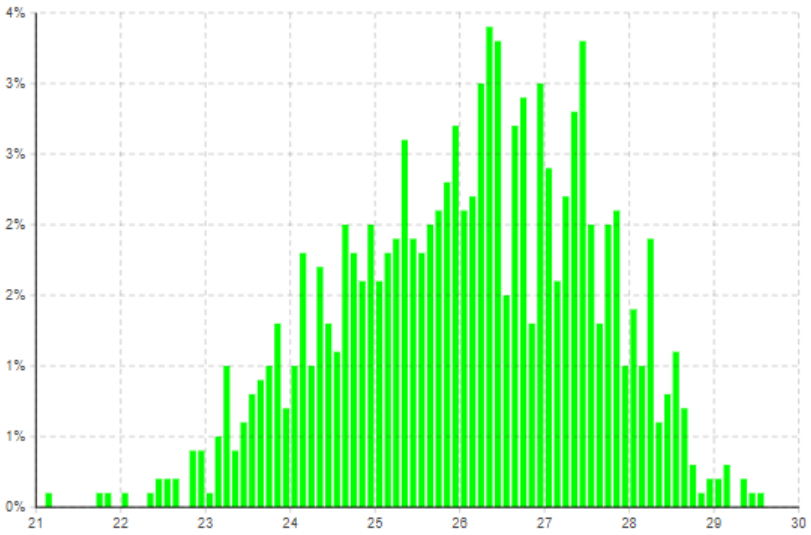
Распределения времени выполнения задач:

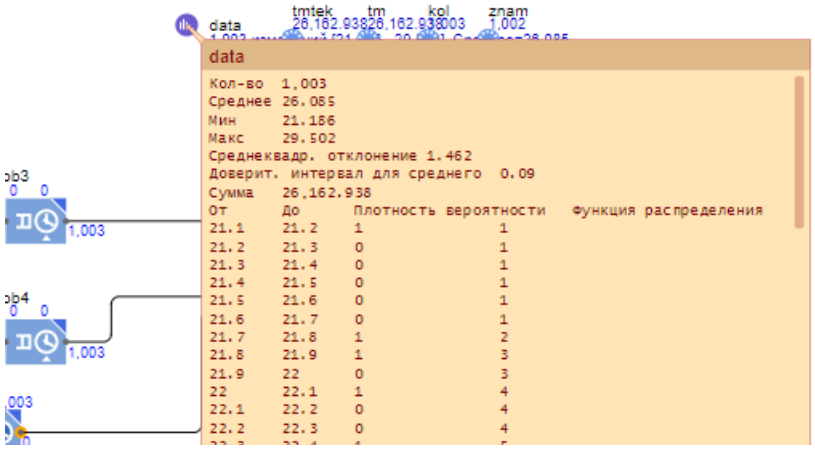
время в часах	Задача 1	Задача 2	Задача 3	Задача 4	Задача 5
a	5	10	12	2	36
b	12	22	17	8	91
m	7	15	13	4	52

Критический путь:

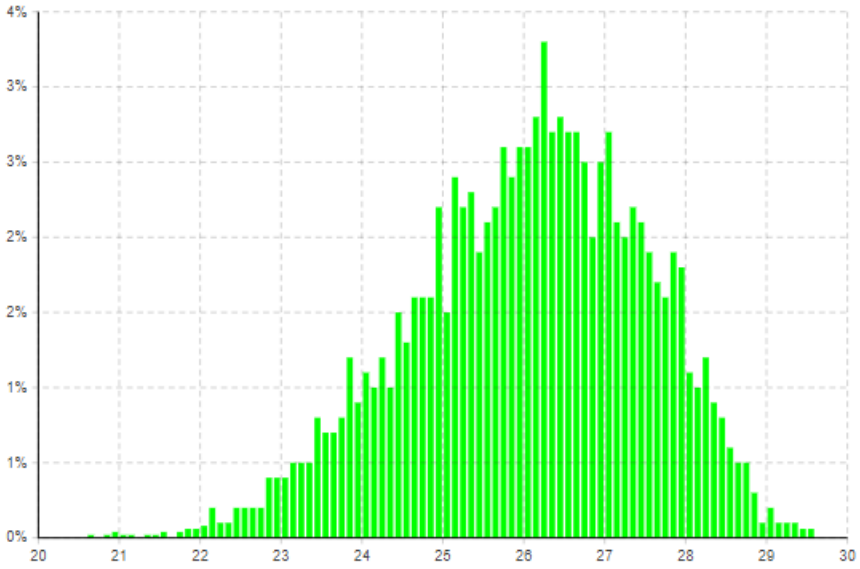
Критический путь проекта	время в часах
41	a
103	b
59	m

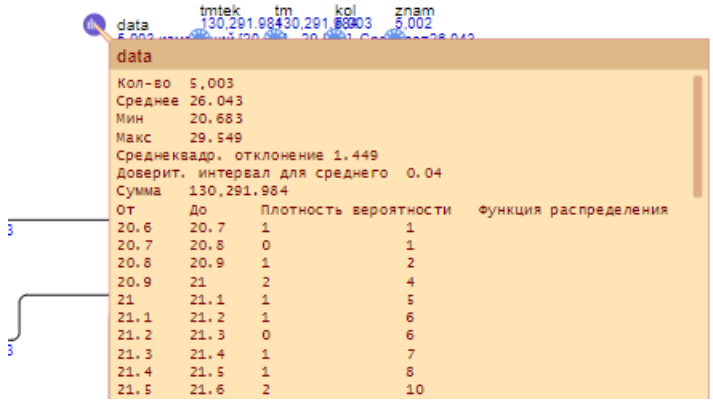
<u>Эксперимент 10</u> – вариант усредненных средних значений, 1001 повторений
Кол-во повторений – 1001





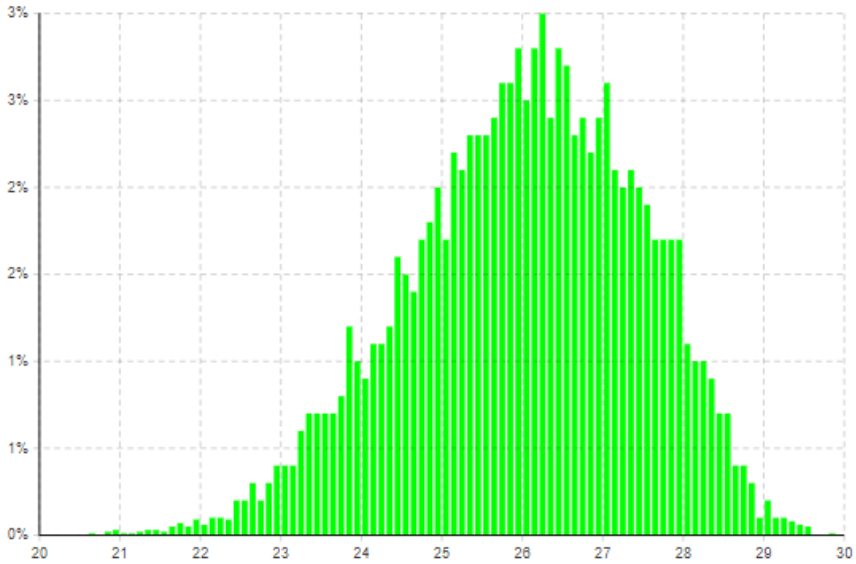
Эксперимент 11 – вариант усредненных средних значений, 5001 повторений
Кол-во повторений – 5001

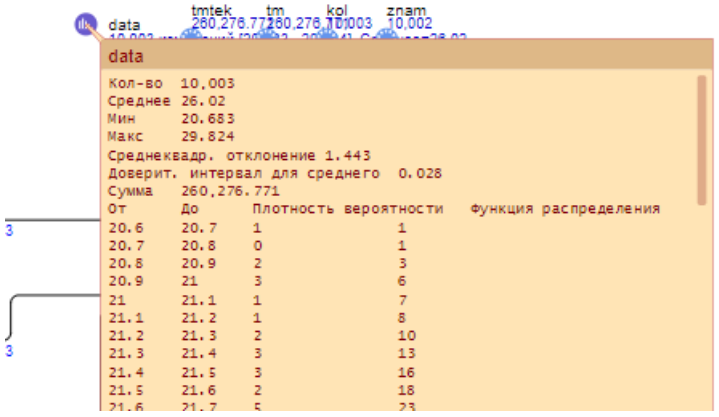




Эксперимент 12 – вариант усредненных средних значений, 10001 повторений
Кол-во повторений – 10001
Критический путь:

Критический путь проекта	время в часах
112	a
281	b
91	m





Имитация полностью разошлась с ручным расчётом: критический путь по модели значительно занижен. Это свидетельствует о том, что при малом числе задач дисбаланс распределений усиливается и модель нуждается в калибровке.

эксперименты 10-12				
	1001	5001	10001	ручной
мин	21	20	20	41
макс	29	29	29	103
м	26	26	26	59

Эксперимент 13 – вариант средних значений сдвинутых к минимуму, 1001 повторений

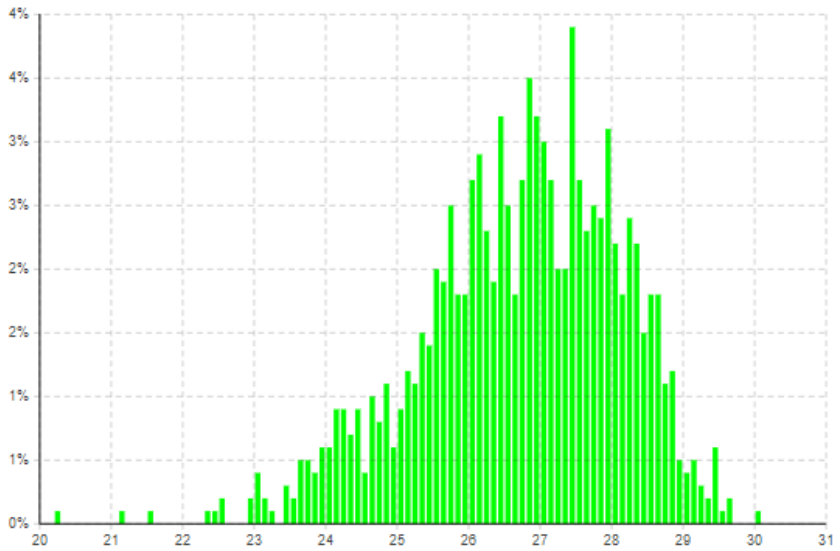
Вид – параллельно-последовательный
Кол-во задач – 5
Вероятность выполнения работ с 1 раза – 100 %
Кол-во повторений – 1001

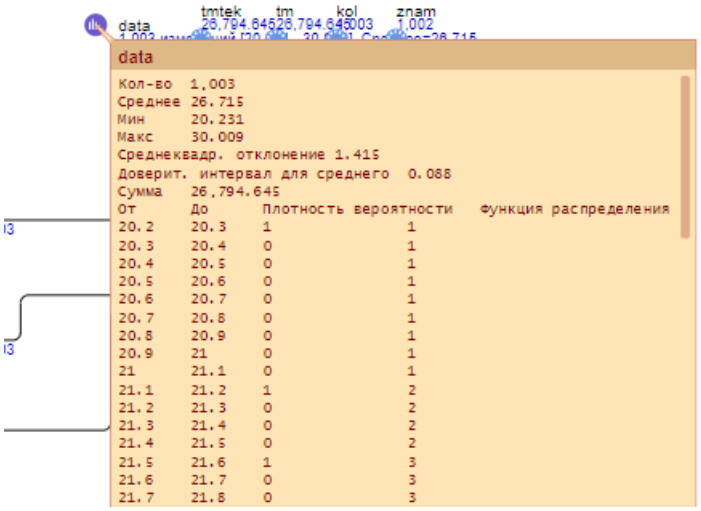
Распределения времени выполнения задач:

время в часах	Задача 1	Задача 2	Задача 3	Задача 4	Задача 5
a	5	10	12	2	36
b	12	22	17	8	91
m	6	12	13	3	40

Критический путь:

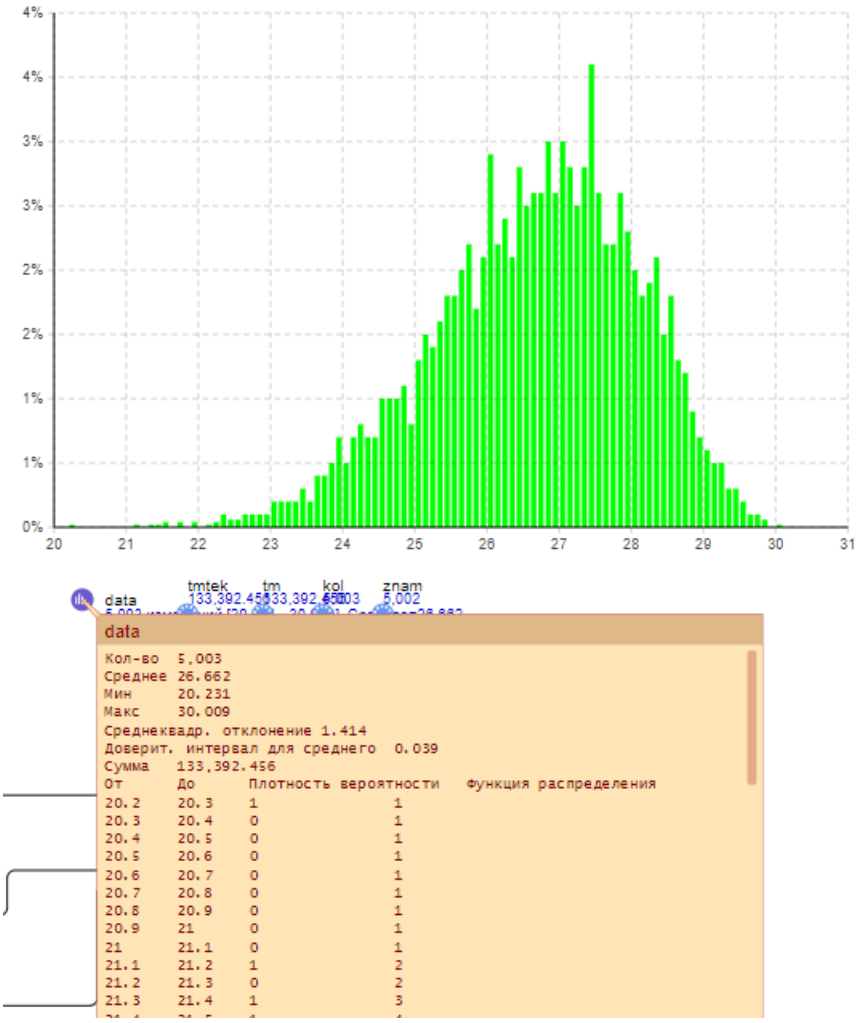
Критический путь проекта	время в часах
41	a
103	b
49	m



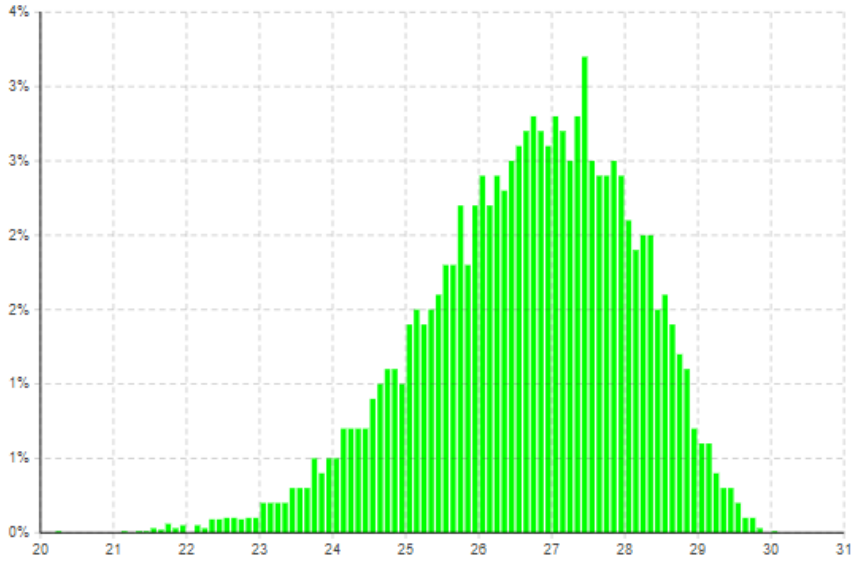


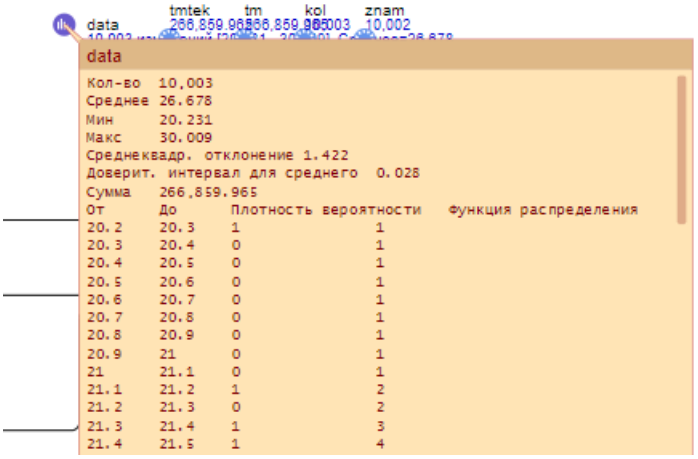
Эксперимент 14 – вариант средних значений сдвинутых к минимуму, 5001 повторений

Кол-во повторений – 5001



Эксперимент 15 – вариант средних значений сдвинутых к минимуму, 10001 повторений
Кол-во повторений – 10001





Графики фиксируют сдвиг вправо, однако значения по-прежнему расходятся с ручными расчётами. Наблюдается занижение критического пути и слабая чувствительность к числу итераций.

эксперименты 13-15				
	1001	5001	10001	ручной
мин	20	20	20	41
макс	30	29	29	103
м	26	26	26	49

Эксперимент 16 – вариант средних значений сдвинутых к максимуму, 1001 повторений

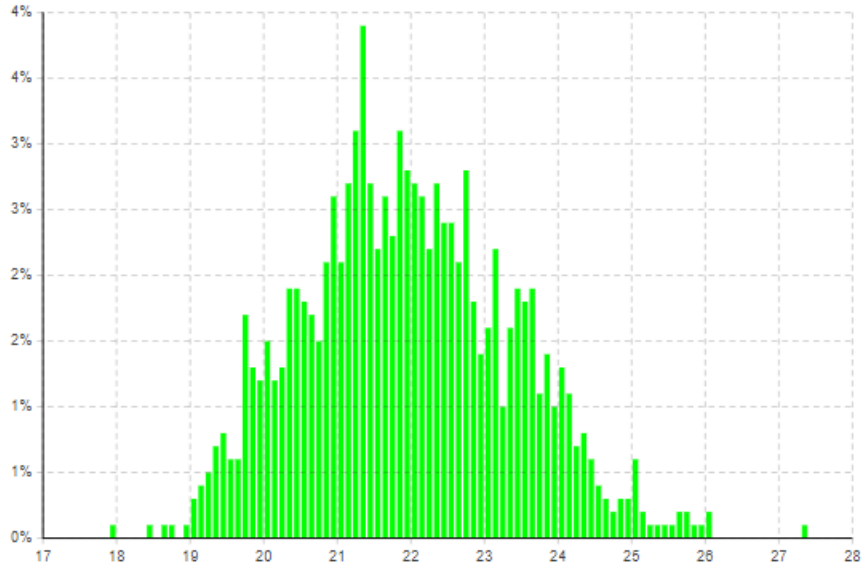
Вид – параллельно-последовательный
Кол-во задач – 5
Вероятность выполнения работ с 1 раза – 100 %
Кол-во повторений – 1001

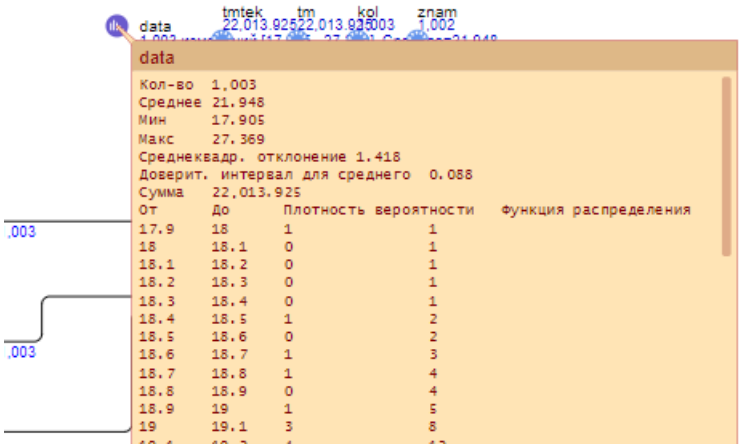
Распределения времени выполнения задач:

время в часах	Задача 1	Задача 2	Задача 3	Задача 4	Задача 5
a	5	10	12	2	36
b	12	22	17	8	91
m	10	20	16	7	85

Критический путь:

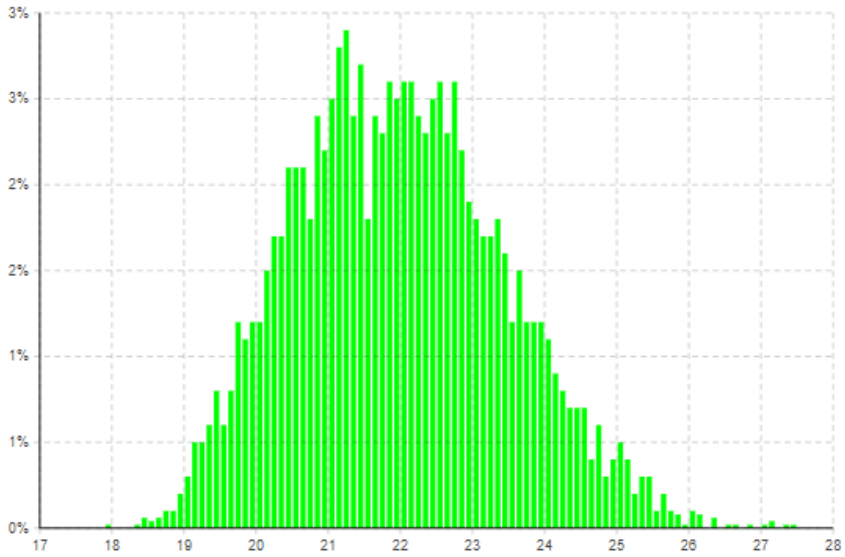
Критический путь проекта	время в часах
41	a
103	b
95	m

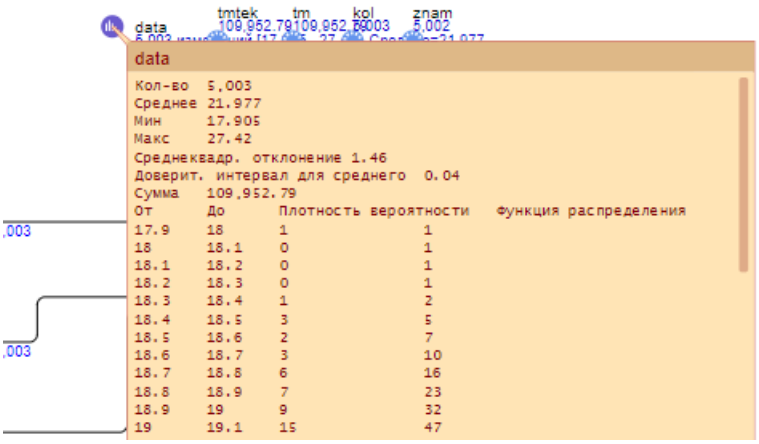




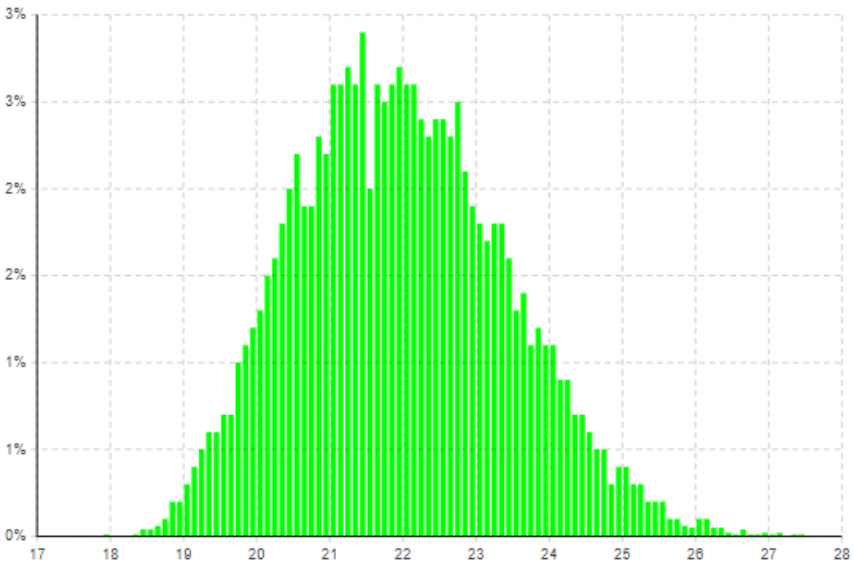
Эксперимент 17 – вариант средних значений сдвинутых к максимуму, 5001 повторений

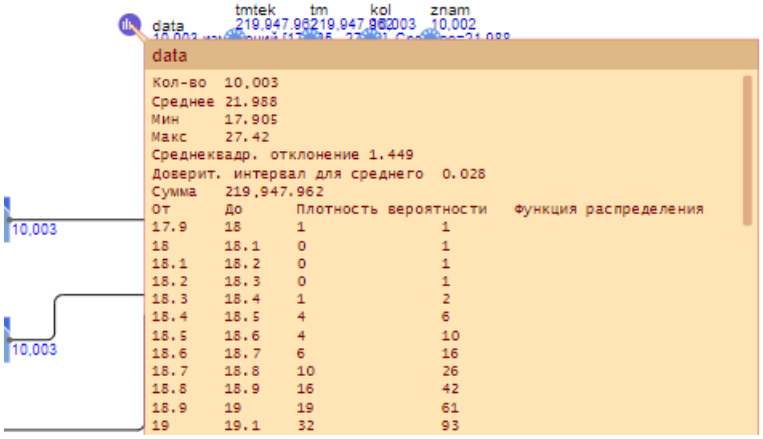
Кол-во повторений – 5001





Эксперимент 18 – вариант средних значений сдвинутых к максимуму, 10001 повторений
Кол-во повторений – 10001

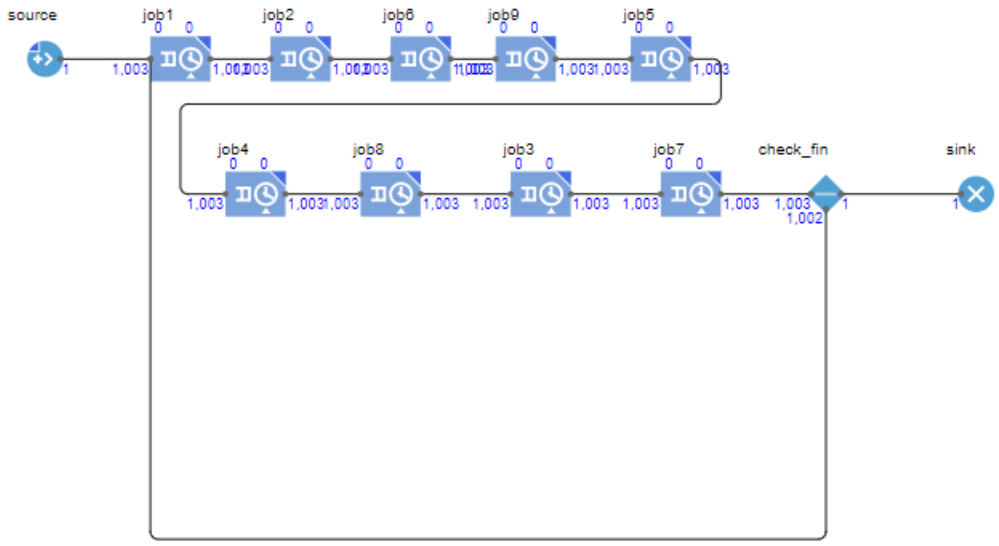




Отмечен сдвиг влево и устойчивое занижение критического пути относительно аналитики. Итоговые значения демонстрируют схожую картину с предыдущей группой — слабое соответствие модели ручным подсчётам.

эксперименты 16-18				
	1001	5001	10001	ручной
мин	17	17	17	41
макс	27	27	27	103
м	21	21	21	95

Последовательный вариант, 9 работ



Вид работ – последовательные
Кол-во работ – 9
Вероятность выполнения работ с 1 раза – 100 %

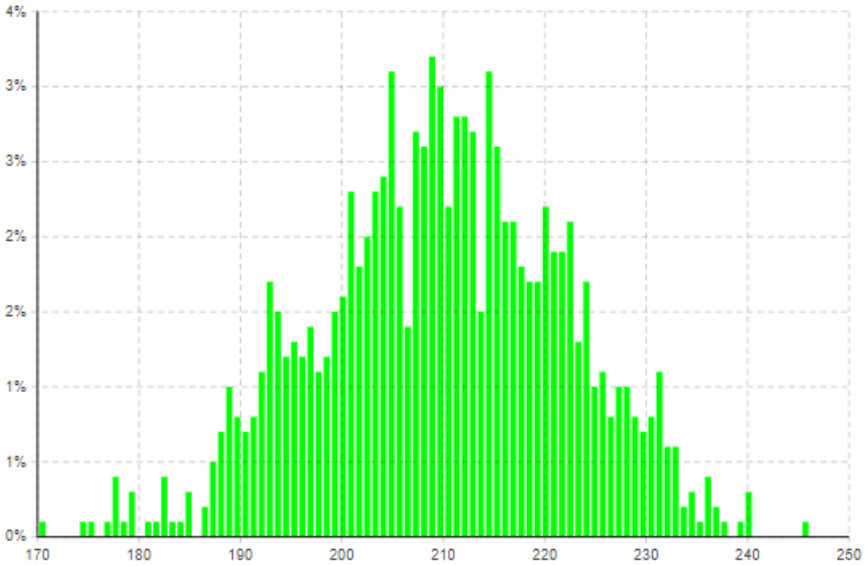
Распределение времени выполнения работ:

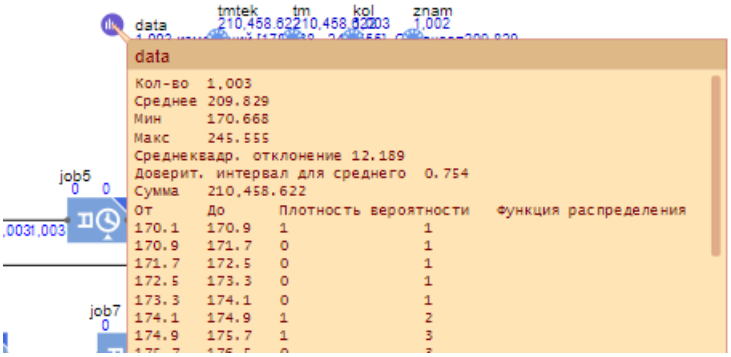
время в часах	Задача 1	Задача 2	Задача 3	Задача 4	Задача 5	Задача 6	Задача 7	Задача 8	Задача 9
a	5	10	12	2	36	10	1	22	14
b	12	22	17	8	91	36	4	58	33
m	7	15	13	4	52	21	3	37	24

Критический путь:

Критический путь проекта	время в часах
112	a
281	b
176	m

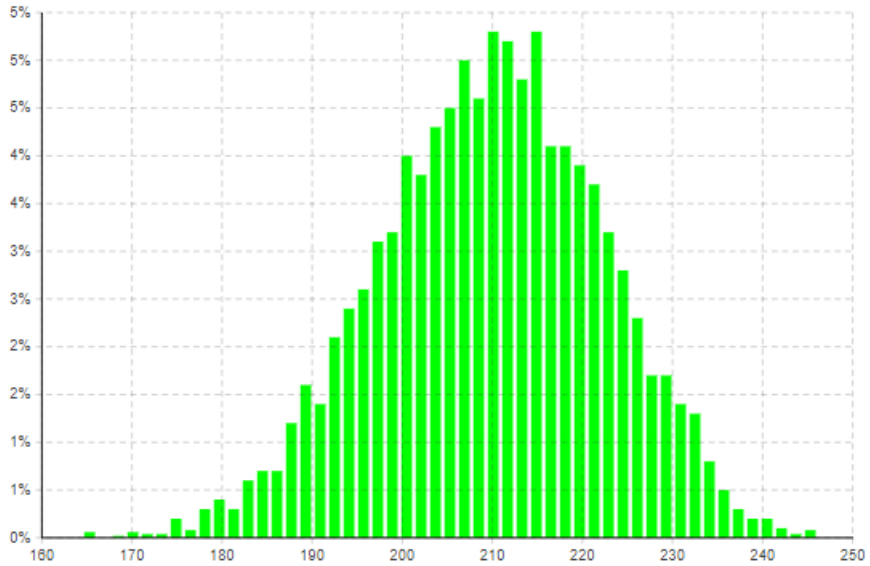
Эксперимент 19 – средние значения, 1001 повторений
Кол-во повторений – 1001

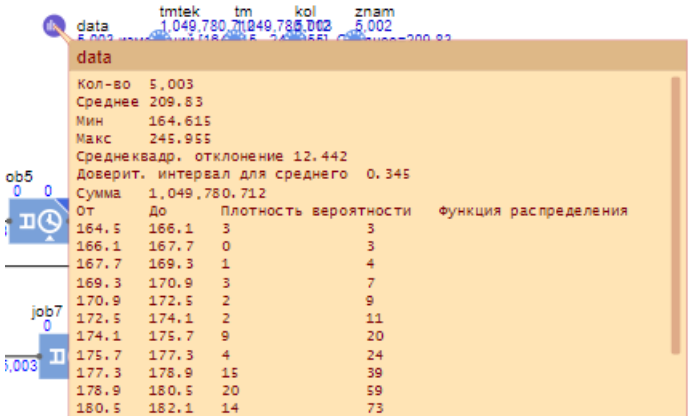




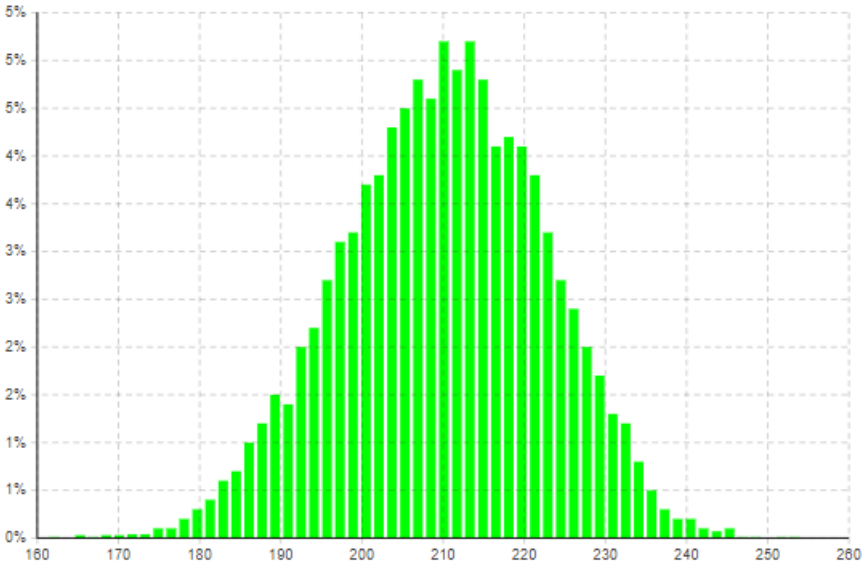
Эксперимент 20 – средние значения, 5001 повторений

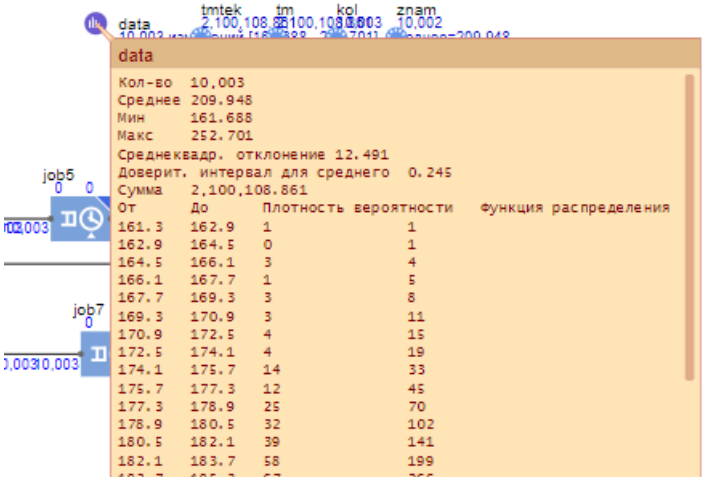
Кол-во повторений – 5001





Эксперимент 21 – средние значения, 10001 повторений
Кол-во повторений – 10001





Совпадение с ручными расчётами есть только по максимальным значениям, минимальные и средние существенно завышены. Это подтверждает, что линейная структура усиливает расхождения.

эксперименты 19-21				
	1001	5001	10001	ручной
мин	170	164	161	112
макс	245	245	252	281
м	209	209	209	176

Эксперимент 22 – средние значения сдвинуты к минимальным, 1001 повторений

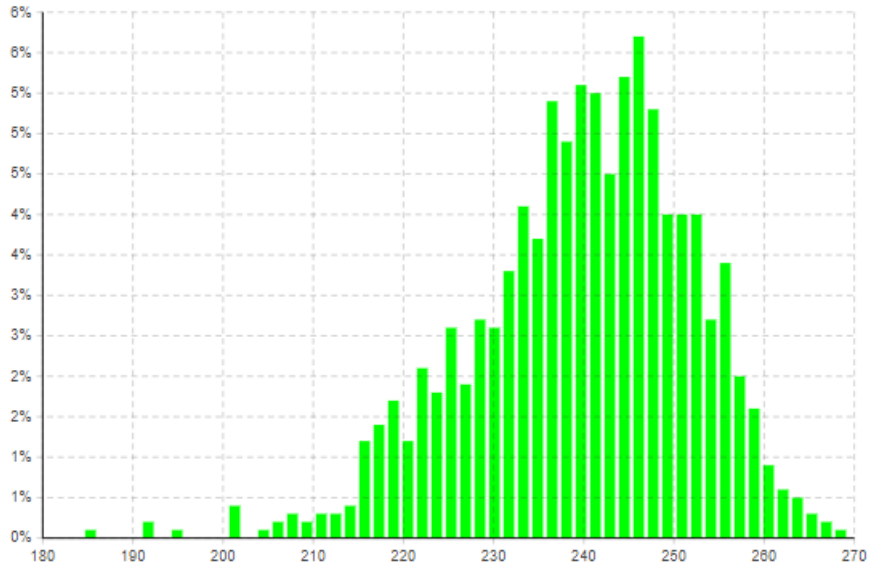
Кол-во повторений – 1001

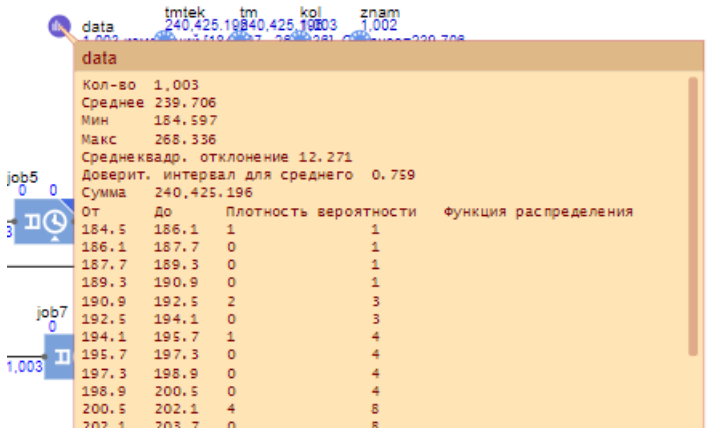
Распределение времени выполнения работ:

время в часах	Задача 1	Задача 2	Задача 3	Задача 4	Задача 5	Задача 6	Задача 7	Задача 8	Задача 9
a	5	10	12	2	36	10	1	22	14
b	12	22	17	8	91	36	4	58	33
m	6	12	13	3	41	13	2	25	17

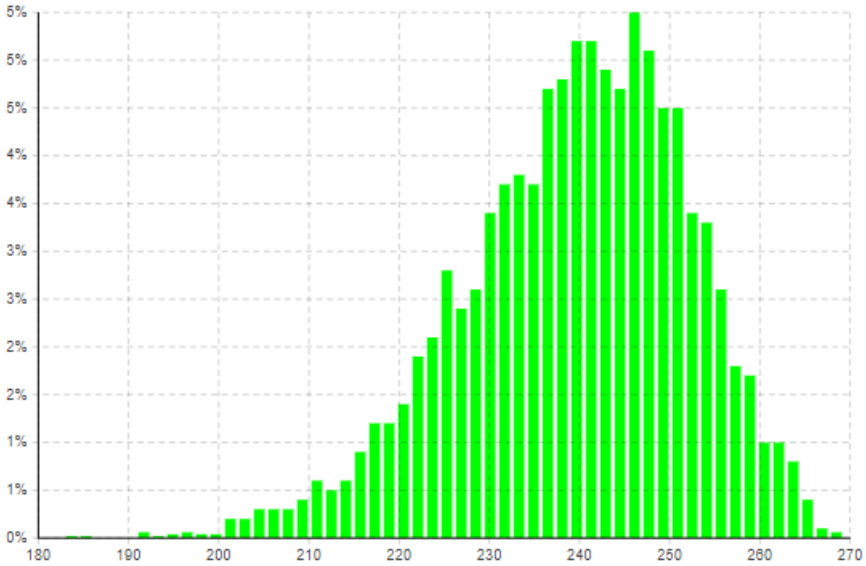
Критический путь:

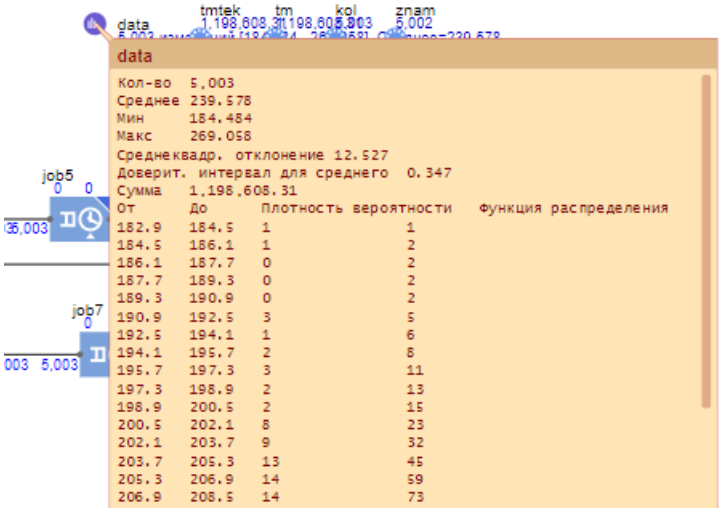
Критический путь проекта	время в часах
112	a
281	b
132	m



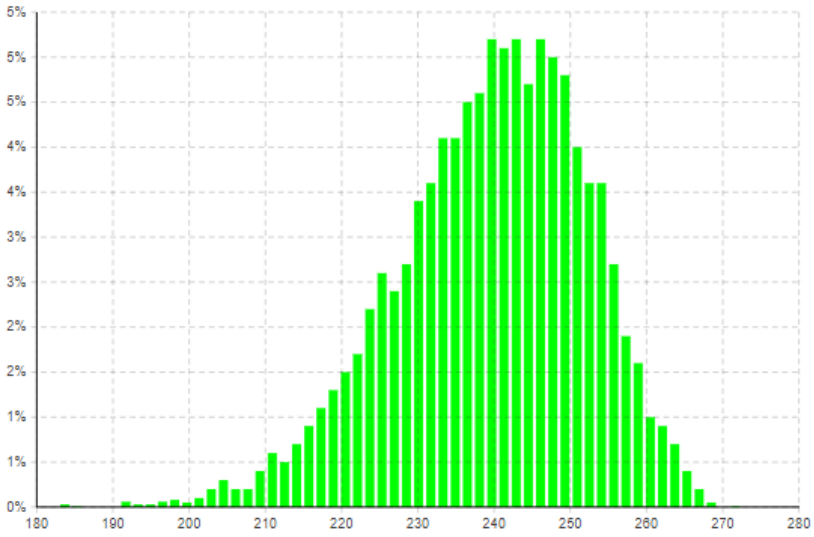


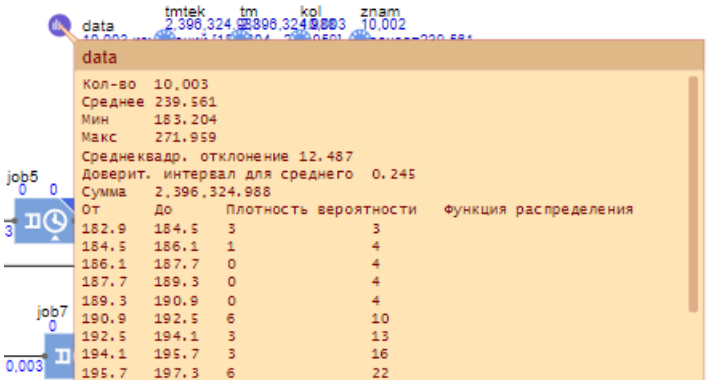
Эксперимент 23 – средние значения сдвинуты к минимальным, 5001 повторений
Кол-во повторений – 5001





Эксперимент 24– средние значения сдвинуты к минимальным, 10001 повторений
Кол-во повторений – 10001





Графики сдвигаются вправо, совпадение снова фиксируется только по максимальным значениям. Модель демонстрирует избыточные значения по средним и минимальным сценариям.

эксперименты 22-24				
	1001	5001	10001	ручной
мин	184	184	183	112
макс	265	269	271	281
м	239	239	239	132

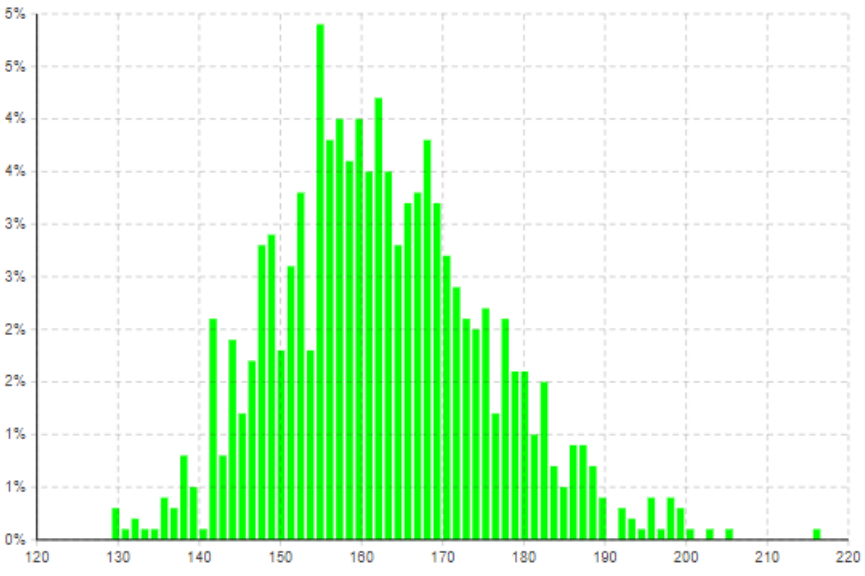
Эксперимент 25 – средние значения сдвинуты к максимальным, 1001 повторений
Кол-во повторений – 1001

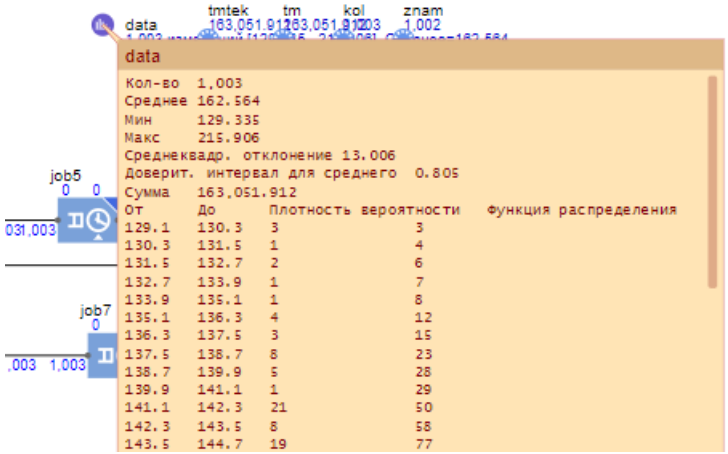
Распределение времени выполнения работ:

время в часах	Задача 1	Задача 2	Задача 3	Задача 4	Задача 5	Задача 6	Задача 7	Задача 8	Задача 9
a	5	10	12	2	36	10	1	22	14
b	12	22	17	8	91	36	4	58	33
m	10	19	16	7	85	29	3	49	30

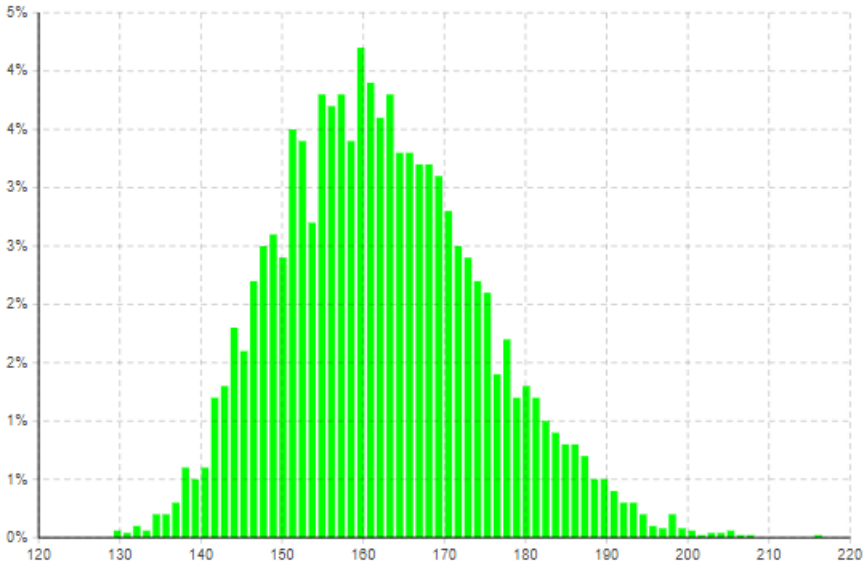
Критический путь:

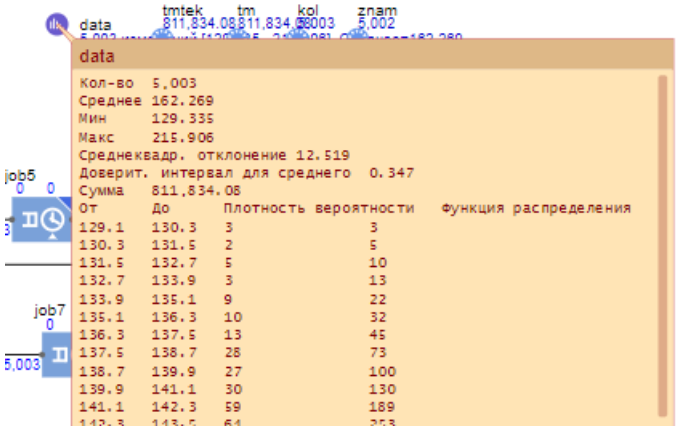
Критический путь проекта	время в часах
112	a
281	b
248	m



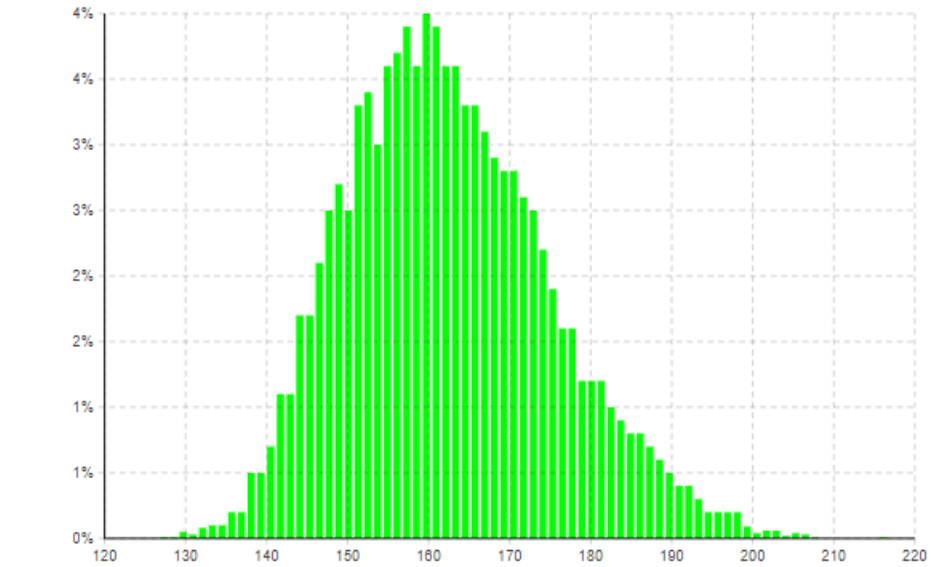


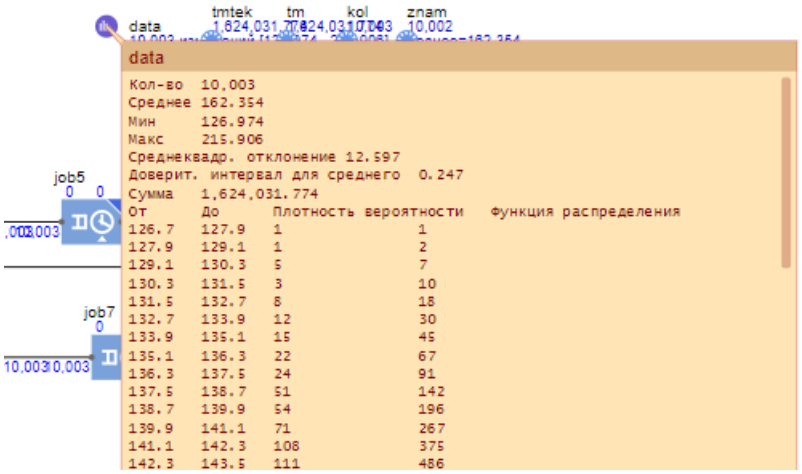
Эксперимент 26 – средние значения сдвинуты к максимальным, 5001 повторений





Эксперимент 27 – средние значения сдвинуты к максимальным, 10001 повторений
Кол-во повторений – 10001





Здесь отмечено совпадение по минимальным значениям, но при этом максимальные и средние значительно расходятся с ручными данными. Смещение к максимуму улучшает нижние оценки, но снижает точность верхних.

эксперименты 25-27				
	1001	5001	10001	ручной
мин	129	129	126	112
макс	215	215	215	281
м	162	162	162	248

ПРИЛОЖЕНИЕ Г – Расчеты полная версия

Рассмотрим один из экспериментов на следующем графике, описывающем последовательность выполнения задач (рисунок. Г.1).

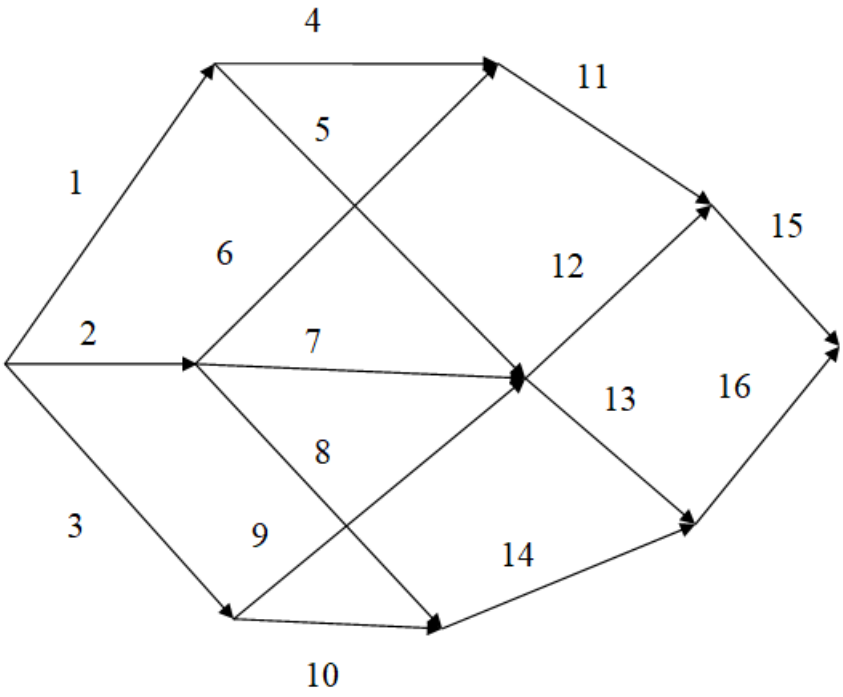


Рисунок Г.1 – Пример сетевого графика

На весах графа показаны идентификаторы задач. Пусть, без ограничения общности, имеются 4 исполнителя для выполнения данных задач.

По формуле 2.9 рассчитаем ориентировочную длительность выполнения каждым исполнителем каждой из задач. Результате представлены в Таблица. Г.1.

Таблица Г.1 - Оценки длительностей выполнения каждой из задач каждым исполнителем

Задача	Исполнит. 1	Исполнит. 2	Исполнит. 3	Исполнит. 4
1	6,667	8,000	7,667	5,833
2	8,667	6,833	7,667	7,000
3	9,333	7,167	10,000	9,000
4	7,000	9,167	7,833	7,000
5	8,000	8,833	9,000	7,833
6	7,167	9,000	8,000	7,167
7	9,833	7,833	9,000	8,000
8	10,000	8,000	9,167	7,833
9	5,167	6,000	8,000	4,833
10	5,833	6,833	6,833	5,167
11	7,833	6,833	7,167	6,000
12	8,000	7,000	7,000	7,000
13	9,000	7,833	8,167	8,667
14	7,167	9,000	8,167	7,667
15	6,167	8,000	7,833	6,000
16	7,833	9,000	9,833	7,000

На предварительном этапе планирования определяется раннее, позднее время начала задач и их резерв. При этом следует отметить, что, с учетом того, что пока назначение исполнителей не было выполнено, оценки длительностей пока неизвестны. В связи с этим, воспользуемся формулой 2.11, если известно некоторое вероятностное распределение исполнителей по задачам или формулой 2.12, если такой информации нет. Предположим, что проект достаточно новый, и какая-либо информация о вероятности назначения той или иной задачи каждому из специалистов ответствует. Тогда для временных оценок воспользуемся формулой среднего значения, в результате чего получим следующие предварительные характеристики (Таблица Г.2).

Таблица Г.2 - Предварительные временные характеристики для исследуемого сетевого графика

Задача	средняя длит	T_i раннее	T_i позднее	резерв
1	7,042	0	0,748	0,748
2	7,542	0	0	0
3	8,875	0	1,333	1,333
4	7,750	7,042	11,334	4,292
5	8,417	7,042	7,791	0,748
6	7,834	7,542	11,250	3,708
7	8,667	7,542	7,542	0
8	8,750	7,542	7,875	0,332
9	6,000	8,875	10,208	1,333
10	6,167	8,875	10,458	1,583
11	6,958	15,375	19,084	3,709
12	7,250	16,208	18,792	2,584
13	8,417	16,208	16,208	0
14	8,000	16,292	16,625	0,332
15	7,000	23,458	26,042	2,584
16	8,417	24,625	24,625	0

В данной таблице во втором столбце приведена ориентировочная длительность выполнения каждой задачи, в третьем – раннее время решения задачи, в четвертом – позднее время решения. Последний столбец показывает временной резерв задачи, который потребуется в случае, когда необходимо делать отбор тех задач, которые надо выполнить в данный момент, и тех, которые следует перенести на более поздний период. Данная информация является весьма грубой, и необходима лишь для того, чтобы ориентироваться в первоначальных временных резервах, которые в дальнейшем будут уточняться, во-первых, при планировании работ, во-вторых, при уточнении времени их завершения и последующей коррекции.

Далее будем итеративно проходить по всем временным интервалам и назначать исполнителей, согласно алгоритму, представленному на рис. 3.3.

На первом шаге можно выполнить задачи 1,2 и 3, т.к. все остальные работы будут последующими. Приведем временные характеристики данных задач для каждого из исполнителей (Таблице Г.3).

Таблица Г.3 - Временные значения для первых трех работ

Задача	Исполнит. 1	Исполнит. 2	Исполнит. 3	Исполнит. 4
1	6,667	8,000	7,667	5,833
2	8,667	6,833	7,667	7,000
3	9,333	7,167	10,000	9,000

Построим ориентировочную матрицу выигрыша для каждой задачи j в случае, если бы ей не был назначен исполнитель i. Для этого найдем, как указано в алгоритме для каждого значения максимум из оставшихся элементов в строке (временные затраты другого исполнителя – наилучшего для данной задачи) и в столбце (временные затраты данного исполнителя при выполнении другой задачи) и вычтем двойное значение временных затрат данного исполнителя для данной задачи. В частности, для матрицы, приведенной в Таблице Г.3, такая матрица выигрыша будет иметь вид, приведенный в Таблице Г.4.

Таблица Г.4 - Матрица выигрыша

Задача	Исполнит. 1	Исполнит. 2	Исполнит. 3	Исполнит. 4
1	1,166	-3,334	-1,834	2,001
2	-3,834	0,501	-0,834	-1,334
3	-4,832	1,499	-5,166	-5

Найдем наибольшее значение в данной матрице и соответствующие строку и столбец. Назначим исполнителю 4 задачу 1, вычтем из матрицы первую строку и четвертый столбец.

Таблица Г.5.

Задача	Исполнит. 1	Исполнит. 2	Исполнит. 3
2	8,667	6,833	7,667
3	9,333	7,167	10,000

Сформировав для данной таблица аналогичную матрицу, получим.

Таблица Г.6.

Задача	Исполнит. 1	Исполнит. 2	Исполнит. 3
2	-1,168	1,168	1,499
3	-2,832	1,832	-5,166

Назначим исполнителю 2 задачу 3. Оставшуюся задачу делает исполнитель, у которого значение целевой функции лучше, в этом случае это исполнитель 3. Таким образом, после первого шага получим следующее закрепление (Таблица Г.7).

Таблица Г.7 - Закрепление задач за специалистами после первого шага алгоритма

Специалист	Задача	Вр нач	Вр оконч
1	-		
2	3	0	7,167
3	2	0	7,667
4	1	0	5,833

Для следующего шага уточним длительность выполнения каждой задачи закрепленным исполнителем и скорректируем временные показатели. Выпишем задачи, которые можно выполнить на этом шаге и определим время начала каждой задачи как максимум из времени завершения предыдущей задачи и освобождения исполнителя (Таблица Г.8).

Таблица Г.8 - Время завершения задач каждым из специалистов

Задача	1	2	3	4
4	5,833+7,000	5,833+9,167	5,833+7,833	5,833+7,000
5	5,833+8,000	5,833+8,833	5,833+9,000	5,833+7,833
6	7,667+7,167	7,667+9,000	7,667+8,000	7,667+7,167
7	7,667+9,833	7,667+7,833	7,667+9,000	7,667+8,000
8	7,667+10,000	7,667+8,000	7,667+9,167	7,667+7,833
9	7,167+5,167	7,167+6,000	7,167+8,000	7,167+4,833
10	7,167+5,833	7,167+6,833	7,167+6,833	7,167+5,167

Расставим приоритеты задачам согласно их временному резерву, приведенному в Таблице Г.2 (Таблица Г.9).

Таблица Г.9 - Приоритизация задач

Работа	1	2	3	4	приор
4	12,833	15	13,666	12,833	7
5	13,833	14,666	14,833	13,666	3
6	14,834	16,667	15,667	14,834	6
7	17,5	15,5	16,667	15,667	1
8	17,667	15,667	16,834	15,5	2
9	12,334	13,167	15,167	12	4
10	13	14	14	12,334	5

Будем на данном шаге планировать назначение задачам 5,7,8,9, а остальные перенесем на следующий период (Таблица Г.10).

Таблица Г.10 - Временные значения выбранных в приоритет задач по исполнителям

Задача	1	2	3	4
5	13,833	14,666	14,833	13,666
7	17,5	15,5	16,667	15,667
8	17,667	15,667	16,834	15,5
9	12,334	13,167	15,167	12

Результат решения и финальное закрепление задач за исполнителями после шага два отражены в Таблице Г.11.

Таблица Г.11 - Закрепление задач за специалистами после второго шага алгоритма

Исполнитель	Задача	Время начала	Время окончания
1	5	5,83	13,83
2	7	7,67	15,50
3	8	7,67	16,83
4	9	7,17	12,00

После назначения исполнителей задачам 5, 7, 8, 9 из ближайших задач у нас остались задачи 4, 6, 10, 12 и 13 (т.к. завершены все работы для их начала) со следующими приоритетами (Таблица Г.12):

Таблица Г.12 - Ближайшие задачи с приоритетами

Задача	1	2	3	4	slack	приор
4	24,67	23,837	24,503	19	4,292	5
6	24,837	23,67	24,67	19,167	3,708	4
10	23,503	21,503	23,503	17,167	1,583	2
12	25,67	21,67	23,67	19	2,584	3
13	25,67	24,50	24,83	25,33	0	1

На третьем шаге последовательно определяя приоритеты и свободных исполнителей будем закреплять специалистов за задачами 4, 6, 10 и 13. Подробное решение по шагам можно посмотреть в приложении В. В Таблице Г.13 отражен результат закрепления исполнителей за задачами.

Таблица Г.13 - Закрепление задач за специалистами после третьего шага алгоритма

Исполнитель	Задача	Время начала	Время окончания
1	4	13,83	20,83
2	13	15,50	23,33
3	6	16,83	24,83
4	10	12,00	17,17

На четвертом шаге будем определять исполнителей для задач 11, 12, 14 (Таблица Г.14).

Таблица Г.14 - Закрепление задач за специалистами после четвертого шага алгоритма

Исполнитель	Задача	Время начала	Время окончания
1	14	20,83	28,00
2	12	23,33	30,33
3	-	-	-
4	11	20,83	26,83

На пятом (финальном шаге) закрепим исполнителей для задач 15 и 16 (Таблица Г.15).

Таблица Г.15 - Закрепление задач за специалистами после пятого шага алгоритма

Исполнитель	Задача	Время начала	Время окончания
1	15	30,33	36,50
2	-	-	-
3	-	-	-
4	16	28,00	35,00

Когда все задачи распределены в соответствии с приоритетами и для каждой задачи закреплен исполнитель можно вернуться к подсчету критического пути.

Критический путь, посчитанный по обобщенному среднему времени до приоритизации и распределения, отражен в Таблице Г.16.

Таблица Г.16 - Критический путь посчитанный по обобщенному среднему времени до приоритизации и распределения

Задача	средняя длит	Ti раннее	Ti позднее	резерв
1	7,042	0	0,748	0,748
2	7,542	0	0	0
3	8,875	0	1,333	1,333
4	7,75	7,042	11,334	4,292
5	8,417	7,042	7,791	0,748
6	7,834	7,542	11,25	3,708
7	8,667	7,542	7,542	0
8	8,75	7,542	7,875	0,332
9	6	8,875	10,208	1,333
10	6,167	8,875	10,458	1,583
11	6,958	15,375	19,084	3,709
12	7,25	16,208	18,792	2,584
13	8,417	16,208	16,208	0
14	8	16,292	16,625	0,332
15	7	23,458	26,042	2,584
16	8,417	24,625	24,625	0

На критическом пути лежат 4 задачи (2-7-13-16), а значение критического пути равно 26,042.

Критический путь, который получен в результате проведенных по алгоритму расчетов, представлен в Таблице Г.17.

Таблица Г.17 - Критический путь полученный после расчетов по алгоритму

Исполнитель	Задача	Длительность	ES	EF	LS	LF	Slack
4	1	5,833	0,00	5,83	2,34	8,17	2,34
3	2	7,667	0,00	7,67	0,00	7,67	0,00
2	3	7,167	0,00	7,17	3,67	10,83	3,67
1	4	7	5,83	12,83	12,83	19,83	7,00
2	5	8,833	5,83	14,67	8,17	17,00	2,34
2	6	9	7,67	16,67	10,83	19,83	3,17
3	7	9	7,67	16,67	8,00	17,00	0,33
1	8	10	7,67	17,67	7,67	17,67	0,00
4	9	4,833	7,17	12,00	12,17	17,00	5,00
3	10	6,833	7,17	14,00	10,83	17,67	3,67
4	11	6	16,67	22,67	19,83	25,83	3,17
3	12	7	16,67	23,67	18,83	25,83	2,17
2	13	7,833	16,67	24,50	17,00	24,83	0,33
1	14	7,167	17,67	24,83	17,67	24,83	0,00
4	15	6	23,67	29,67	25,83	31,83	2,17
4	16	7	24,83	31,83	24,83	31,83	0,00

Критический путь состоит из 4 задач – 2-8-14-16 и составляет 31,83. При значении критического пути, посчитанного по среднему времени, 26,042. Результат сравнения представлен в Таблице Г.18.

Таблица Г.18 - Результат сравнения длин и значений критического пути полученного разными способами.

вариант решения	кол-во работ на критическом пути	список работ на критическом пути	время	соотношение
по средним значениям работ	4	2-7-13-16	26,042	100,00%
математический (сбитые приоритеты)	4	2-8-14-16	31,83	122,23%

В результате расчета критического пути по алгоритму получена более точная оценка длительности проекта, которая поможет точнее запланировать требуемое время на выполнение проекта и не выйти за рамки обозначенных в начале проекта сроков.

Г.2. Коррекция

Помимо штатного хода выполнения проекта, в процессе реализации могут возникать внештатные ситуации, оказывающие влияние на сроки и ресурсы. К их числу относятся, например, увольнение либо временная нетрудоспособность сотрудника, назначенного на выполнение определённого стека задач. Аналогично, возможна ситуация обнаружения ошибки в ранее выполненной задаче, что потребует её повторного решения для устранения выявленных недостатков. Кроме того, в ходе приёмочных встреч заказчик может вносить новые требования, дополняющие изначально согласованный стек задач.

Для минимизации вероятности возникновения внезапных дополнительных задач лицо, принимающее решения (ЛПР), до начала работ над проектом должно обеспечить разработку, согласование и подписание технического задания (ТЗ). Данный подход не исключает полностью появления новых задач, однако позволяет вывести их за рамки первоначально согласованного объёма работ и оформить как отдельную категорию — задания на изменения (ЗНИ). Это предотвращает бесконтрольное внесение правок, поскольку качество выполнения задач и их объём изначально закреплены документально.

Далее рассмотри работу алгоритма в случае появления коррекции.

Предположим, что выполнив порученную ему задачу на шаге 3 (таблица Г.13) исполнитель 4 заболел и стал недоступен для выполнения работ. Пересчитаем альтернативное окончание проекта начиная с шага 4, если доступно только 3 исполнителя – 1, 2 и 3.

Результат шага четыре с учетом коррекции отражен в Таблице Г.19

Таблица Г.19 -Обновленное закрепление задач за исполнителями на шаге 4 с учетом коррекции.

Исполнитель	Задача	Время начала	Время окончания
1	14	20,83	28,00
2	11	20,83	30,16
3	12	23,33	31,83

Если сравнить время окончания в Таблицах Г.14 и Г.19, то видно, что время окончания задачи 14 не изменилось, а для задач 11 и 12 увеличилось в связи с изменением исполнителей из-за произведенной корректировки.

Теперь с учетом отсутствия исполнителя 4 перераспределим задачи 15 и 16 на шаге 5. Результат представлен в Таблице Г.20.

Таблица Г.20 - Обновленное закрепление задач за исполнителями на шаге 5 с учетом коррекции.

Исполнитель	Задача	Время начала	Время окончания
1	16	28	35,83
2	-	-	-
3	15	31,83	38,16

Из таблице выше видим, что обновленное критическое время проекта равно 38,16.

Дополним результирующую Таблицу Г.18 новым результатов (Таблица Г.21).

Таблица Г.21 - Обновленная таблица сравнения различных вариантов решения задачи

вариант решения	кол-во работ на критическом пути	список работ на критическом пути	время	соотношение
по средним значениям работ	4	2-7-13-16	26,042	100,00%
По алгоритму	4	2-8-14-16	31,83	122,23%
По алгоритму с коррекцией	4	2-8-14-16	38,16	146,53%