Федеральное государственное автономное образовательное учреждение
высшего образования
«Национальный исследовательский университет
«Высшая школа экономики»

На правах рукописи

Марон Максим Аркадьевич

МОДЕЛИ, МЕТОДЫ И АЛГОРИТМЫ УПРАВЛЕНИЯ КОНТРОЛЕМ КАЧЕСТВА В ДИНАМИКЕ СЛОЖНЫХ ПРОЕКТОВ

ДИССЕРТАЦИЯ

на соискание ученой степени кандидата наук

по прикладной математике НИУ ВШЭ

Научный руководитель:
доктор технических наук, профессор
Акопов Андраник Сумбатович

Москва 2019

Оглавление

Введение

Актуальность темы

В современной быстро меняющейся бизнес-среде непрерывно возрастает роль проектного управления в деятельности компаний. Требования к продукту, получаемому при реализации проекта, будут соблюдены тогда и только тогда, когда все работы проекта выполнены правильно. При выполнении любой работы проекта может возникнуть ошибка, то есть отклонение от установленных требований. Если ошибки в работах будет выявлены только на этапе проверки соответствия конечного продукта установленным требованиям, то потребуется много времени для их локализации. Сроки проекта могут оказаться сорванными.

Для парирования данного риска необходимо предусмотреть контрольные точки, в которых будут выполнены промежуточные проверки. Проблема разработки моделей, методов и алгоритмов управления динамикой сложных проектов путём осуществления проверок в рационально выбранных контрольных точках является актуальной. Её решение является темой данного диссертационного исследования.

Целесообразной является стратегия своевременного обнаружения ошибок в работах проекта, при которой каждая из работ проверяется на соответствие важнейшим параметрам (частичная проверка), а полные проверки осуществляются после некоторых работ – контрольных точек, выбираемых при планировании проекта.

Степень разработанности темы

Работы, посвящённые управлению проектами, не дают ответа на вопрос о том, как и в каком количестве выбирать контрольные точки в проектах. Под диагностикой проектов в существующей литературе по управлению проектами понимается в первую очередь определение показателей, позволяющих установить, что произошло отклонение от базового плана,

требующее перепланирования или принятия других управленческих решений.

Значительную роль в решении этих вопросов играют работы Аньшина В. М., Богданова В. В., Джаафари А., Ильина В. В., Товба А. С., Царькова И. Н., Ципеса Г. Л., De Marco A., Thakurta R.

Вместе с тем имеются чёткие рекомендации по снижению вероятности возникновения ошибок в работах для особо ответственных проектов. Это пооперационный контроль. Для проектов разработки программных продуктов разработана методология объектно-ориентированного программирования, которая существенно упрощает локализацию ошибок с помощью контрольных точек. Здесь необходимо особо отметить работы Авдошина С. М., Крука Е. А., Липаева В. В.

Наиболее полно вопросы проверки правильности функционирования технических и программных систем, а также вопросы локализации неисправностей решаются в науке, названной «техническая диагностика». Здесь основополагающими являются работы Пархоменко П. П. и его учеников.

Несмотря на наличие явной аналогии между функциональной моделью непрерывной технической системы и сетевым графиком, проекты и технические системы являются принципиально различными объектами диагноза. Анализ возможности применения к проектам методов оптимизации процесса поиска неисправностей показал следующее. В наибольшей степени методы оптимизации развиты в технической диагностике применительно к задаче построения оптимальных условных алгоритмов поиска неисправностей. Построению оптимальных алгоритмов поиска неисправностей посвящены работы Карибского В. В., Пархоменко П. П., Согомоняна Е. С., Халчева В. Ф. Из регулярных математических методов оптимизации для построения оптимальных условных алгоритмов поиска неисправностям применяются метод

динамического программирования и метод ветвей и границ. При современном уровне развития вычислительной техники регулярные методы не могут гарантировать построение оптимального алгоритма поиска неисправностей для технических систем, в которых число возможных неисправностей больше, чем 25–30.

Построение оптимального набора контрольных точек является ещё более сложной задачей, которая проработана значительно слабее. В известной степени это связано с тем, что в первую очередь развивались техническая диагностика дискретных устройств и соответственно методы построения тестов.

Объектом исследования являются проекты.

Предметом исследования является выбор контрольных точек для своевременного обнаружения ошибок в работах проекта.

Целью исследования является разработка методов выбора контрольных точек для своевременного обнаружения ошибок в работах проекта.

Для достижения поставленной цели сформулированы следующие задачи исследования:

1. Разработать диагностическую модель проекта. Эта модель должна давать возможность установить связь между ошибками в работах и результатами проверок.
2. Разработать математическую модель возникновения ошибок в работах, которая позволит рассчитывать вероятности ошибок.
3. Предложить критерии для сравнения вариантов расстановки контрольных точек в зависимости от того, как потери от задержки окончания проекта, вызванные ошибками в работах, зависят от времени восстановления правильности его выполнения.
4. Разработать методы определения наборов контрольных точек в соответствии с предложенными критериями и экспериментально оценить их

эффективность, если это будут эвристические, а не регулярные методы, гарантирующие нахождение оптимума.

5. Разработать алгоритмы и программное обеспечение для экспериментальной проверки эффективности предложенных методов и их практического применения к расстановке контрольных точек в реальных проектах.

В диссертационной работе впервые получены следующие результаты, характеризующиеся научной новизной:

1. Предложена диагностическая модель проекта (ДМП). Она позволяет представить соответствие между результатами проверок и возможными ошибками в работах в виде таблицы возможных ошибок (ТВО) проекта, для построения которой можно использовать методы, разработанные в теории графов. Она является основой для решения поставленной задачи определения наборов контрольных точек в проекте

2. Разработана математическая модель возникновения ошибок в работах проекта. Модель соответствует реальностям процесса возникновения ошибок, если: ошибки возникают независимо друг от друга; при реализации проекта не допускается ослабления контроля качества выполнения работ даже по мере приближения запланированных сроков его окончания; к недобросовестным исполнителям своевременно применяются меры воздействия, заставляющие их соблюдать установленные требования; нет оснований предполагать, что одни исполнители работают лучше других. С помощью предложенной модели можно рассчитать вероятности не только для одиночных ошибок, но и для любой их комбинации. Предложена и модель расчёта вероятностей ошибок для ООП. В этом частном, но исключительно важно случае, выбор варианта расстановки контрольных точек уместно проводить в предположении о наличии ошибки в одной работе проекта.

3. Предложен критерий для сравнения наборов контрольных точек в ООП для случая, когда потери от задержки проекта нелинейно зависят от времени восстановления правильности выполнения проекта.

4. Предложен метод определения количества и мест расположения контрольных точек, который можно применить к большинству типовых проектов. В нём учитывается как возможность наличия ошибок в нескольких работах, так и затраты времени на выполнение операций контроля.

5. Предложены методы выбора контрольных точек в ООП, причём и для мультисценарных проектов, где возможно изменение состава работ. Методы используют, предложенные автором: диагностическую модель проекта (ДМП) и модель для расчёта вероятностей ошибок в ООП. Методы не являются регулярными, но их научной базой является теория информации К. Шеннона, применение которой даёт хорошие результаты при планировании экспериментов, а проверки в контрольных точках являются именно экспериментами, выполняемыми в целях диагностирования проекта. Экспериментально доказана эффективность предложенных методов.

Теоретическая и практическая значимость. Теоретическая значимость определяется тем, что впервые предложен регулярный подход к решению проблемы поиска ошибок в работах проекта. Разработанная модель возникновения ошибок в проекте, основанная на предположении о простейшем потоке ошибок, может быть дополнена на случаи, когда нарушается стационарность потока ошибок по мере приближения окончания проекта, а также на случай, когда возникает последействие. Предложенные методы выбора контрольных точек могут быть обобщены на случаи, когда результаты проверок не являются абсолютно достоверными. Практическая значимость состоит в том, что руководителю проекта даётся реальный инструмент для определения необходимого числа и мест проведения промежуточных проверок с целью своевременного обнаружения ошибок в

работах проекта. Результаты данной работы успешно внедрены для использования в компанию IBS, что подтверждается актом о внедрении.

Методология и методы исследования

Исследования основаны на: математических моделях проекта, теории информации К. Шеннона, методах теории графов, изложенных в работах Алескерова Ф. Т., методах теории надёжности и теории массового обслуживания, изложенных в работах Гнеденко Б., Беляева Ю., Каштанова В. А., Соловьёва А., современных технологиях программирования, изложенных в работах Авдошина С. М., Крука Е. А., Липатова В. В., методах проведения имитационного эксперимента, изложенных в работах Акопова А. С.

Основные положения, выносимые на защиту:

- диагностическая модель проекта;
- математическая модель возникновения ошибок в проектах;
- критерий сравнения наборов контрольных точек в проектах;
- методы расстановки контрольных точек в особо ответственных проектах;
- метод расстановки контрольных точек в мультисценарном проекте;
- метод расстановки контрольных точек в проекте, допускающем появления множества ошибок.

Достоверность изложенных в работе результатов подтверждается теоретическими выкладками, а также результатами численного моделирования и экспериментальных исследований по Методу Монте – Карло.

Апробация работы

По результатам работы сделаны доклады:

- на международной ежегодной научно-практической конференции «НОВОЕ В НАУКЕ И ОБРАЗОВАНИИ на тему: «Промежуточный контроль как средство снижения рисков проектов». МЕИЭФиП. Москва, 2015 г.;
- на международной ежегодной научно-практической конференции «НОВОЕ В НАУКЕ И ОБРАЗОВАНИИ» на тему «Имитационная модель хода выполнения проекта с ошибкой». МЕИЭФиП. Москва, 2016 г.;
- на 12ой ежегодной международной конференции “Corporate Responsibility Research Conference” на тему: “Stability of realization of strategic initiatives for ensuring corporate responsibility”. Istanbul, 2016.;
- на международной ежегодной научно-практической конференции «НОВОЕ В НАУКЕ И ОБРАЗОВАНИИ» на тему: «Диагностика проектов». МЕИЭФиП. Москва, 2017 г;
- На заседании научного семинара «Экспертные оценки и анализ данных». Институт проблем управления РАН. Москва 2019.

Личный вклад. Все представленные в диссертации результаты получены лично автором. При подготовке статей и докладов автор опирался на помощь соавторов и научного руководителя.

Публикации

Основные результаты опубликованы в работах:

- Марон, А. И. Информационный подход к организации контроля проектов / А. И. Марон, М. А. Марон // Бизнес-ИНФОРМАТИКА. –2012. – № 4 (22). – С. 47–53.
- Maron, M. A. The choice of control points of projects taking into account possible change of structure of works / M. A. Maron // Business Informatics. 2016. – № 2 (36). – Pp. 57–62.
- Maron, M. A. Diagnostics of Projects / M. A. Maron // European Research Studies Journal. – 2018. – Vol. 21. – № 1. – Pp. 18–30.

Всего по теме диссертации опубликовано 12 работ, из них 5 работ в журналах, входящих в Перечень ВАК.

Объем и структура работы. Диссертационная работа изложена на 146 страниц (без приложений), включает 4 таблицы, 16 рисунков. Состоит из введения, трех глав, заключения, списка литературы и наименований.

Глава 1 Анализ проработанности проблемы диагностики проектов

1.1 Диагностика проектов – проблема и анализ подходов в литературе по управлению проектами

Рассмотрим проект, в результате которого создаётся уникальный продукт. Этот продукт должен отвечать установленным требованиям [ГОСТ Р ИСО 21500, 2014; PMBoK, 2013; GAPPS, 2006; OGC 2009]. Все требования к продукту будут соблюдены тогда и только тогда, когда все работы проекта выполнены правильно. Однако при выполнении любой работы проекта может возникнуть ошибка, то есть отклонение от установленных требований. Проверка соответствия конечного продукта установленным требованиям является обязательным этапом реализации проекта. Если ошибки в работах будет выявлены только на этом этапе, то потребуется много времени для их локализации, то есть обнаружения работ, выполненных неправильно. На практике подобная ситуация практически неизбежно приводит к срыву сроков завершения проекта и соответствующим материальным потерям. Поэтому необходимо осуществлять проверки правильности выполнения работ в ходе выполнения проекта. Если осуществлять для **каждой** работы проверку соответствия результата **всем** установленным требованиям и исправлять ошибки, то будет гарантировано соответствие конечного продукта проекта всем установленным требованиям, но при этом сроки проекта и его стоимость могут увеличиться до недопустимых значений. Выполнение полной проверки только тогда, когда конечный продукт

получен, и полная проверка всех работ в процессе выполнения проекта — это два крайних варианта. Первый из них оставляет риск срыва сроков проекта при условии возникновения ошибок в работах, а второй неизбежно значительно увеличивает длительность проекта.

Вместе с тем существует более рациональная стратегия обнаружения ошибок в работах проекта, основанная на том, что работы проекта связаны не только связями, определяющими порядок их выполнения во времени, но и результатами. Поясним это на простейшем примере. Допустим, что две работы проекта – A и B – связаны зависимостью «окончание – начало». Работа A имеет один результат, который используется в работе B для получения её результата. Достаточно провести полную проверку результата работы B, чтобы в случае положительного результата сделать вывод, что обе работы выполнены правильно, а в случае отрицательного результата проверки сделать вывод, что как минимум одна ошибка имеет место. Целесообразной является стратегия своевременного обнаружения ошибок в работах проекта, при которой каждая из работ проверяется на соответствие важнейшим параметрам (частичная проверка), а полные проверки осуществляются после некоторых работ, выбираемых при планировании проекта. Набор работ, являющихся объектами полных проверок, будем в дальнейшем называть набором контрольных точек проекта.

Для каждого крупного проекта существует огромное количество возможных наборов контрольных точек. Так, если в проекте n работ и решено создать m контрольных точек, то число возможных наборов контрольных точек равно числу сочетаний C_n^m. Процесс контроля правильности выполнения работ и локализации ошибок уместно назвать диагностикой проекта. Рациональный выбор контрольных точек является одной из важнейших задач такой диагностики.

Определим, к какой группе процессов управления проектами относится диагностика проекта, в том смысле как она определена выше. Согласно

PMBoK, риск проекта – это неопределенное событие или условие, наступление которого оказывает положительное или отрицательное влияние на одну или несколько целей проекта, которыми являются качество, расписание, стоимость и содержание. Проектов без рисков нет. Это следует из свойства уникальности. Различным аспектам управления рисками проектов посвящено большое количество работ, посвящённых как общетеоретическим вопросам [Артемьев Д., Гергерт Т., Пономарёва Т., 2013; Казак А. Ю., 2013; Копылова О. В., 2013; Кузьмин Е. А., 2012; Тебеньков А. Н., 2012], так и вопросам управления рисками при реализации определённых типов проектов [Авдошин С. М., Песоцкая Е. Ю., 2008 и 2015; Магомаева Л., 2013; Махтева И. П., 2013; Патрушева А. А., 2013; Песелис А., 2012; Coombs C. R., 2015; Futrell R., Shafer D., Shafer L., 2002; Taylor H. 2006].

Возникновение ошибок при выполнении работ является риском. Риски делятся по степени влияния на проект на положительные, отрицательные [Ципес Г. Л., Товб А. С., 2009]. Положительные риски – это события, при возникновении которых характеристики проекта изменятся в лучшую сторону. Так, можно назвать положительным риском то событие, что заказчик проекта выделит дополнительную сумму денег по собственной инициативе без изменений остальных условий выполнения проекта и дополнительных обращений. В такой ситуации у компании, выполняющей данный проект, при прочих равных условиях увеличится бюджет на работу. Отрицательный риск – это событие, при возникновении которого характеристики проекта изменяются в худшую сторону. К данной категории можно отнести ситуацию, когда бюджет проекта, предполагающего закупку комплектующих за рубежом, уменьшается вследствие падения курса национальной валюты. Возникновение ошибок при выполнении работ проекта – это отрицательный риск.

Определим, к какому методу управления рисками проектами относится локализация ошибок с помощью контрольных точек. Существуют следующие четыре стратегии реагирования на риски.

1. Уклонение

Уклонение от риска совершается путем изменения плана управления проекта, тем самым устраняется влияние угрозы на проект. Самым радикальным уклонением от риска будет прекращение проекта.

2. Передача

Передача риска осуществляется путем передачи риска третьей стороне и избегания, таким образом, ответственности за риск. Практически всегда выплачивается премия третьей стороне за принятие ответственности за риск на себя. Примером передачи риска может быть оформление страховки.

3. Снижение

Снижение риска – это план действий, в результате которого уменьшается вероятность наступления риска или его влияние на результаты проекта. Этот метод предполагает увеличение работы и проведение дополнительных мероприятий.

4. Принятие

Если все способы реагирования на риск не эффективны или экономически не выгодны, то риск принимается и никаких действий не осуществляется до его наступления, а уже при столкновении с ним вырабатывается стратегия реагирования.

Локализация ошибок с помощью контрольных точек является одним из методов снижения влияния риска возникновения ошибок в работах на результаты проекта.

Отказ от промежуточного контроля является принятием риска. Возникает вопрос: сколько должно быть контрольных точек и как их выбрать

для того, чтобы снижение последствий реализации риска позднего обнаружения ошибок в работах проекта было выгоднее его принятия.

Анализ показывает, что ответа на этот вопрос в литературе по управлению проектами нет. Действительно, под диагностикой проектов в существующей литературе по управлению проектами понимается в первую очередь определение показателей, позволяющих установить, что произошло отклонение от базового плана, требующее перепланирования или принятие других управленческих решений [Джаафари А., Джабари Н., 2008; Матяш И. В., 2013].

Расчёт стандартных показателей отклонения по стоимости и времени реализован в MS – Project и других программах управления проектами, которые можно считать CASE – средствами, соответствующими PMBoK [Чатфилд К., Джонсон Д., 2013; Трофимов В. В. и др., 2006; Biafore B., 2010]. Близкими по целям к таким работам являются статьи, посвящённые устойчивости проектов. В них предложены показатели, методы и алгоритмы, позволяющие определить момент, когда в проекте произошли уже столь серьёзные отклонения, что он не может быть выполнен с заданными ограничениями по времени и стоимости [Аньшин В. М., 2013; Ананьин В. И., 2005; Зуйков К. А., 2012; Ильина О., 2012; Cioffi D. F., 2006; De Marco A., Briccarello D., Rafele C., 2009; Herroelen W., Leus R., 2004; Leus R., Herroelen W., 2004; Sakka O., Barki H., Côté L., 2016].

Другим направлением исследований, которые следует проанализировать на возможность применения предложенных в них методов для решения рассматриваемой проблемы, являются работы, посвящённые обеспечению качества в проектах [ИСО/ТО 10006:1997; Ильин В. В., 2006; Фатрелл Р, Шафер Д., Шафер Л, 2003; Лесных В. В., Литвин Ю. В., 2013; Thakurta R., 2013] Здесь наиболее близкой по тематике можно считать статью, в которой предложен метод определения работ, результаты которых следует контролировать на соответствие по качеству в первоочередном порядке. Этот

метод состоит в следующем. Составляется список всех n работ проекта. Если решено провести контроль результатов m из них на предмет соответствия требованиям по качеству, то выбирается параметр отбора: длительность, трудоёмкость или стоимость. Работы упорядочиваются по этому параметру и для контроля выбираются первые m работ из упорядоченного списка. Если применить аналогичный подход для выбора контрольных точек с целью локализации ошибок, то заведомо будет потеряна информация, заложенная в логических связях между работами.

Вопросам снижения вероятностей рисков вообще и ошибок при выполнении проектов, в частности [Кравченко Т. К., 2013; Соколов М. Ю., Маслова С. В., 2013; Воробьева О. А., 2012; Swartz S. M., 2008], посвящено значительно больше работ, чем проблеме снижения последствий их реализации. Это связано с тем, что для снижения вероятностей рисков применяются в первую очередь организационные меры, а вопросы менеджмента являются доминирующими в работах по управлению проектами [Башмачникова Е. В., Абрамова Л. А., 2013; Трухановский О. М., 2012; Богданов В., 2016; Володин С. В., 2013; Гурин Е. В., Недовесов М. В., 2012; Рудакова А. А., 2012].

Наиболее значимыми для решения задачи выбора контрольных точек являются следующие известные результаты.

1. Эффективным методом снижения вероятности возникновения ошибок при выполнении работ является пооперационный контроль [Якимович Б. А., Коршунов А. И., Кузнецов А. П., 2007].

2. Эффективным методом упрощения поиска ошибок в программах является объектно ориентированное программирование [Липаев В. В., 2006; Васильев А. О., 2012; Златопольский Д., 2012].

Пооперационный контроль применяется тогда, когда результатом проекта должен стать особого ответственный продукт или, как принято говорить в оборонной отрасли, изделие. Приведём пример. Проект направлен

на создание особо ответственного изделия. Одним из этапов проекта является его сборка. До её начала разрабатывается подробная инструкция по сборке. В выполнении работы по сборке участвуют три специалиста. Один зачитывает пункт инструкции, посвящённый очередной операции. Второй специалист выполняет операцию. Третий проверяет правильность её выполнения.

Если пооперационный контроль выполняется для всех работ проекта, то, решая задачу выбора контрольных точек, с высокой степенью достоверности можно предполагать, что кратные ошибки отсутствуют. На практике пооперационный контроль применяется только при реализации особо ответственных проектов из-за того, что требует значительных затрат временных и материальных ресурсов.

Объектно-ориентированное программирование предполагает разбиение программного продукта на модули, обменивающиеся результатами. При этом существенно упрощается локализация ошибок в отдельных модулях.

Стратегия обнаружения ошибок в работах проекта, при которой каждая из работ проверяется на соответствие важнейшим параметрам, а полные проверки осуществляются после некоторых работ, выбираемых при планировании проекта, является типовой для проектов разработки сложных программ [Филипс Дж., 2005]. При этом под «работой» понимается законченный модуль программы. Так, каскадная модель с интеграцией гибких итеративных методов, широко применяемая в настоящее время для разработки программ, предполагает следующие этапы создания программного продукта:

- определение требований;
- проектирование;
- конструирование, называемое также «реализация» либо «кодирование»;
- интеграция;
- тестирование и отладка;

• инсталляция;

• поддержка.

Ошибки на этапах определения требований и проектирования неизбежно приводят к провалу проекта. Соответственно для каждого из этих этапов обязательной является полная проверка. Она происходит при согласовании требований и проектировании структуры будущего программного продукта. При согласовании данных этапов заказчик лично подтверждает, что они выполнены правильно.

Наиболее опасным этапом в плане появления ошибок является конструирование. Поэтому всегда проводится тестирование разработанного модуля, при котором проверяется его реакции на специальные тесты. Однако полное тестирование каждого модуля, как правило, не проводится, поскольку оно требует слишком много времени, проверяются лишь основные реакции. Вместе с тем очередной разрабатываемый модуль использует результаты некоторых из ранее созданных модулей. Если при его тестировании выявлена неправильная реакция, а проведенная проверка показывает, что этот модуль ошибок не содержит, то делается вывод, что по крайней мере одна ошибка допущена в некотором модуле, который передаёт ему результаты. Наличие структуры программы, разделённой на модули, позволяет запланировать, какие модули будут проходить полное тестирование.

Вопрос о том, какие именно модули следует выбрать для полного тестирования, остаётся открытым.

Изложенное позволяет сделать следующие выводы.

1. При выполнении любой работы проекта может возникнуть ошибка, то есть отклонение от установленных требований. Если ошибки в работах будет выявлены только на этапе проверки соответствия конечного продукта установленным требованиям, то потребуется много времени для их локализации. Сроки проекта могут оказаться сорванными.

2. Целесообразной является стратегия своевременного обнаружения ошибок в работах проекта, при которой каждая из работ проверяется на соответствие важнейшим параметрам (частичная проверка), а полные проверки осуществляются после некоторых работ – контрольных точек, выбираемых при планировании проекта.

3. Возникновение ошибок при выполнении работ проекта следует классифицировать как отрицательный риск. Локализация этих ошибок с помощью контрольных точек является методом снижения влияния риска на результаты проекта.

4. Для каждого крупного проекта существует огромное количество возможных наборов контрольных точек. Рациональный выбор контрольных точек является одной из важнейших задач диагностики проектов.

5. Работы, посвящённые управлению проектами, не дают ответа на вопрос о том, как и в каком количестве выбирать контрольные точки в проектах.

Вместе с тем имеются чёткие рекомендации по снижению вероятности возникновения ошибок в работах для особо ответственных проектов. Это пооперационный контроль. Для проектов разработки программных продуктов разработана методология объектно-ориентированного программирования, которая существенно упрощает локализацию ошибок с помощью контрольных точек.

Проанализируем возможность применения существующих методов технической диагностики для решения задачи выбора контрольных точек в проектах.

1.2 Анализ возможности применения моделей технических объектов диагноза в диагностике проектов

Наиболее полно вопросы проверки правильности функционирования технических и программных систем, а также вопросы локализации неисправностей решаются в науке, названной «техническая диагностика» [Карибский В. В., Пархоменко П. П., Согомонян Е. С., Халчев В. Ф., 1976; Пархоменко П. П., Согомонян Е. С., 1981; Сапожников Вл. В., Сапожников В. В., 2004; Шишмарев В., 2013]. Анализируемую систему при этом принято называть объектом диагноза. Объекты диагноза принято разделять на непрерывные, дискретные и смешанные. Модель объекта диагноза – это совокупность данных об объекте, необходимых для решения поставленной задачи технического диагностирования. При этом используются следующие понятия: объект диагноза – техническая система, состоящая из отдельных элементов.

В процессе эксплуатации системы в её элементах могут возникать дефекты, вызывающие неисправности – отклонения некоторых параметров функционирующего элемента от установленных требований. Серьёзные неисправности приводят к отказу системы. Цель диагностики состоит в том, чтобы:

1) обнаружить появление дефекта по его проявлению в виде неисправности;

2) найти дефектный элемент.

Хотя причиной неисправностей являются дефекты, более распространённой терминологией при описании задач диагностики является следующая.

Первая из указанных выше задач называется контролем исправности, вторая – поиском неисправностей. Именно такой терминологии будем придерживаться далее. Для поиска неисправностей выполняются проверки – эксперименты над объектом диагноза, результаты которых известны как для

исправной системы, так и при возможных неисправностях. Проверки выполняются в определённой последовательности, которая называется алгоритмом диагноза. Если факт наличия неисправности установлен, например, зафиксирован отказ системы, то принято говорить об алгоритме поиска неисправностей. Принято различать условные и безусловные алгоритмы диагноза, последние из которых могут быть с условной и безусловной остановкой. Простейшим с математической точки зрения является безусловный алгоритм с безусловной остановкой. Такой алгоритм состоит из фиксированного числа проверок и независимо от порядка их осуществления имеет фиксированное время выполнения. Безусловные алгоритмы диагноза – тесты – широко применяются для контроля и поиска неисправностей дискретных объектов, которые представляют собой совокупность электронных плат высокой степени интеграции. В условном алгоритме поиска неисправностей каждая следующая проверка выбирается с учётом результата предыдущей. Соответственно каждой неисправности соответствует своё время поиска. Такие алгоритмы можно оптимизировать с учётом критерия, отражающего то, как растут потери при возникновении неисправности системы. Если рост потерь является линейным, то оптимальным является алгоритм, при котором минимально среднее время поиска возможных неисправностей объекта диагноза.

Для построения оптимального условного алгоритма поиска неисправности должны быть заданы:

1) перечень возможных неисправностей;
2) перечень допустимых проверок;
3) вероятности неисправностей;
4) время выполнения каждой допустимой проверки;
5) глубина диагноза, как правило, это перечень групп элементов, объединённых способом устранения, например, объединённых в съёмные блоки;

6) результат каждой допустимой проверки при каждой возможной неисправности.

Общей формой представления соответствия между неисправностями и результатами проверок является таблица неисправностей. Эта таблица размерности $n \times m$, где n – количество возможных неисправностей, а m – количество возможных проверок, строки которой соответствуют возможным неисправностям, а столбцы – допустимым проверкам. На пересечении строки i со столбцом j указывается результат проверки $П^j$ при неисправности e_i. Каждая проверка может иметь 2 или более различных результата.

Сложность составления таблиц неисправностей существенно сдерживало внедрение методов технической диагностики. Эта проблема постепенно решается с развитием методов моделирования сложных технических систем и созданием программного обеспечения, поддерживающего такие методы. Вместе с тем кардинальное уменьшение вычислительных проблем возможно только в случае, если ещё на этапе проектирования технической системы учитываются требования по упрощению поиска неисправностей. Например, система создаётся по блочному принципу. В этом случае для представления непрерывного объекта диагноза принято использовать функциональную модель [Карибский В. В., Пархоменко П. П., Согомонян Е. С., Халчев В. Ф., 1976; Малкин С. В., 2008].

Функциональная модель – это форма представления соответствия между неисправностями и результатами проверок, при которой объект диагноза представляется в виде ориентированного графа. Вершины графа соответствуют возможным неисправностям, а выходящие из них дуги, проверкам - измерении определённых диагностических параметров. Результат проверки параметра на выходе вершины j должен иметь результат, отличный от эталонного, если в объекте имеется неисправность e_j или любая

другая неисправность, соответствующая вершине, из которой есть путь в вершину *j*; в противном случае этот результат должен быть эталонным.

При использовании функциональной модели неисправности и блоки системы (группы элементов, объединённых способом устранения неисправностей) отождествляются между собой. В соответствии с этим можно сказать, что функциональная модель представляет диагностируемую систему в виде графа, вершины которого соответствуют блокам, а дуги путям распространения сигналов. Причём проверка сигнала на выходе блока *j* будет иметь положительный результат (такой же, как в исправной системе) тогда и только тогда, когда он исправен и исправны все блоки, лежащие на путях от начальных вершин к данному блоку.

Таблицу неисправностей объекта диагноза, представленного функциональной моделью, можно получить из матрицы смежности вершин соответствующего ему графа. Упрощается и решение задач построения алгоритмов контроля и поиска неисправностей, а также определения минимального (по числу проверок) набора проверок, достаточного для контроля исправности данного объекта.

Так, установлено, что для контроля исправности необходимо и достаточно осуществить проверки на всех выходных блоках функциональной модели. Контроль здесь рекомендуется осуществлять автоматически с помощью датчиков. Вместе с тем этих проверок, как правило, недостаточно для поиска неисправностей с точностью до неисправного блока. Соответственно, если при контроле зафиксирован факт наличия неисправности, то необходимо осуществить её поиск (локализацию с точностью до блока), реализовав определённый алгоритм поиска неисправностей. Для объекта, состоящего из большого числа блоков, даже применение оптимизированного условного алгоритма поиска не может гарантировать того, что, даже при наиболее вероятных неисправностях, поиск будет завершён в приемлемые сроки. В связи с этим на

функциональной модели дополнительно выбираются определённые блоки для автоматического контроля. Совокупность таких блоков принято называть набором промежуточных контрольных точек [Клюев В. В., Пархоменко П. П., Абрамчук В. Е. и др, 1989; Гурин В. и др., 2011].

Перейдём к нашей задаче выбора контрольных точек в проектах. Сетевая модель является первой и основной математической моделью проекта [Баркалов С., Воропаев В. И., Секлетова Г. И., 2005; Царьков И. Н., 2015]. В соответствии с ней проект представляется в виде ориентированного графа, вершины которого соответствуют работам, а дуги логическим связям между ними. Основной логической связью является связь «Окончание – Начало» (Finish to Start).

Проекты и непрерывные технические системы – это существенно разные объекты диагноза. Главное отличие состоит в том, что проект диагностируется в процессе реализации [Петракова В. А., Сомова А. С., 2012]. Вместе с тем аналогия между сетевой моделью проекта и функциональной моделью технической системы, построенной по блочному принципу, предполагает потенциальную возможность применить ряд методов, разработанных в технической диагностике для решения задач диагностики проектов.

Изложенное позволяет сделать следующие выводы.

1. Наиболее полно вопросы проверки правильности функционирования технических и программных систем, а также вопросы локализации неисправностей решаются в науке, названной «техническая диагностика».

2. Аналогия между сетевой моделью проекта и функциональной моделью технической системы, построенной по блочному принципу, предполагает потенциальную возможность применить ряд методов, разработанных в технической диагностике для решения задач диагностики проектов.

3. Проект диагностируется в процессе реализации. Это главное отличие проектов от технических систем при постановке и решении задач диагностики. Поэтому прямой перенос результатов, полученных для технических систем, как правило, невозможен.

В следующем параграфе будут детально рассмотрены процессы:

- контроля отсутствия ошибок в работах проекта;
- поиска и исправления обнаруженных ошибок в работах проекта – восстановления правильности выполнения проекта.

Будет указано, что необходимо учитывать при решении задачи выбора контрольных точек в проектах.

1.3 Процесс восстановления правильности выполнения проекта

Процессам реализации проекта и отклонениям от плана посвящено значительное количество работ. Среди них особенно следует выделить работы, использующие методы имитационного моделирования [Lyneis J. M., Cooper K. G., Els S. A., 2001; Исаев Д. В., 2014; Ирзаев Г. Х., 2012; Jahangirian M. and other, 2010]. При этом используются как дискретно-событийное моделирование [Kelton W., Sadowski R., 2015], так и методы системной динамики [Акопов А. С., 2014]. Однако процесс восстановления правильности выполнения проекта, необходимый для решения задачи выбора контрольных точек, нуждается в отдельном рассмотрении.

Рассмотрим процесс восстановления правильности выполнения проекта, который начинается, когда в контрольной точке зафиксирован отрицательный результат. Для наглядности будем сопровождать изложение рассмотрением примера.

Пусть сетевой график проекта представляет собой правильно пронумерованный граф. Первую контрольную точку на графе проекта

целесообразно определить при планировании. На рисунке 1.1, где изображён сетевой график условного проекта, это работа с номером 1.

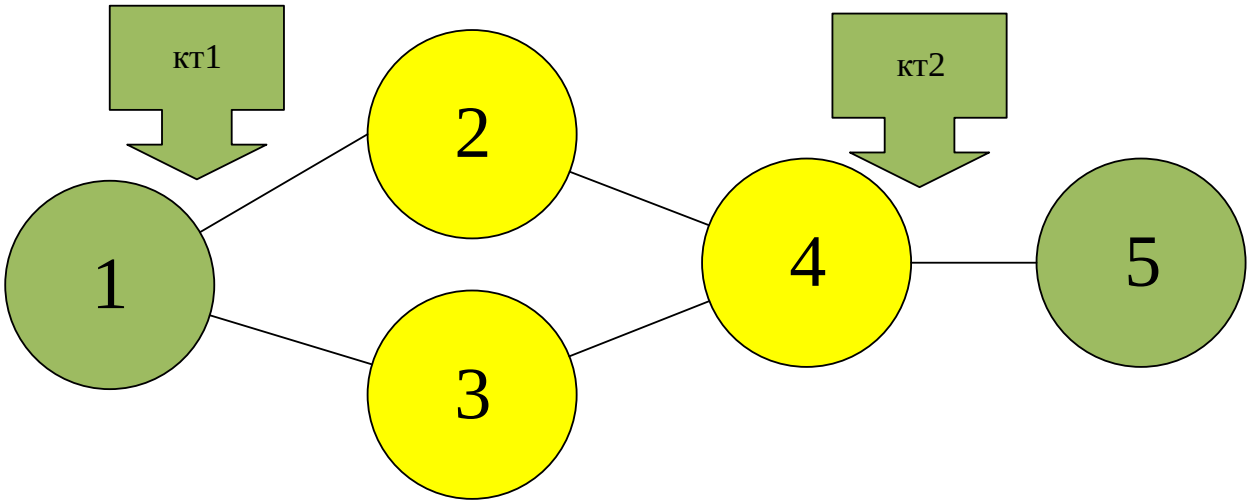

Рис. 1.1. Граф условного проекта с контрольными точками

Если проверка $П^{1}$ дала положительный результат, это означает, что работа 1 выполнена правильно. Выполнение проекта продолжается. Если же эта проверка дала отрицательный результат, то он однозначно свидетельствует о том, что при выполнении работы 1 допущена ошибка. Выполнение последующих работ откладывается. Уполномоченные члены команды проекта устраняют допущенную ошибку. Время t_{yj}, которое придётся затратить на устранение ошибки в работе j, коррелирует с её длительностью t_j. Во многих случаях эта зависимость линейна. В худшем случае время устранения ошибки в работе может занять даже больше времени, чем плановое время выполнения данной работы. Если считать, что исправление ошибки по существу является повторным выполнением работы, то можно принять, что время исправления ошибки в работе равно плановому времени её выполнения.

После устранения ошибки выполнение работ проекта продолжается. Вторая и последующие контрольные точки могут быть определены как при планировании проекта, так и при его реализации. В первом случае при их

выборе придётся основываться на плановой длительности работ и вероятностях ошибок, начально определённых. Во втором случае есть возможность уточнить эти величины с учётом опыта реализации данного проекта. Возможность выбора контрольный точек в процессе реализации проекта принципиально отличает проект, как объект диагноза, от технической системы. В технических системах контрольные точки выбираются или для уже готовой эксплуатируемой системы, или, что более правильно, на этапе её проектирования [Толстяков В. С., 1972; Ефанов Д. В., Плеханов П. А., 2011; Брейдо А., 1983]. В любом случае один раз, а не в динамике.

Продолжим рассмотрение процесса выбора контрольных точек в проекте. Допустим, что второй контрольной точкой назначена проверка результата работы с номером 4.

Если эта проверка даст положительный результат, то можно будет считать, что все параметры работ {2, 3, 4} соответствуют установленным требованиям, как и параметры работы 1, которая проверена и при необходимости исправлена ранее. Если же эта проверка даст отрицательный результат, то это будет означать, что при выполнении как минимум одной из этих работ допущена ошибка. То есть, если зафиксирован отрицательный результат проверки $П^4$, то возможными являются не только ошибки в одной из работ {2, 3, 4}, но и все комбинации ошибок в этих работах – кратные ошибки. Выполнение последующих работ откладывается и запускается процесс восстановления правильности выполнения проекта.

Этот процесс может быть организован по-разному в зависимости от вида проекта.

Рассмотрим проекты, для которых выполняется следующее условие. Если работа А является предшественником работы В, то результат работы А используется для получения результата работы В. Будем называть проекты, обладающие таким свойством проектами с работами, связанными по

результатам (ПСР). Заметим, что в классической математической модели проекта, разработанной в пятидесятых годах прошлого века и используемой поныне, понятие результата отсутствует. Это понятие было чётко зафиксировано в стандарте IDEF0, который возник позднее и предназначен для описания бизнес-процессов. В соответствии с этим стандартом результат (Output) является обязательным атрибутом работы наряду с управлением (Control) [Маклаков С. В., 2005]. В соответствии с IDEF0, если результат работы A является входом работы B, то этот результат перерабатывается (изменяется) для получения результата работы B. Аналогичное предположение используется в моделях ARIS [Davis R., 2004]. Именно так будем понимать в дальнейшем связь по результатам между работами проекта. При этом возможны следующие типовые варианты, отличные по виду алгоритмов восстановления.

Вариант 1 характерен для проектов создания программных комплексов, состоящих из отдельных модулей, реализующих заложенные в них методы, при разработке которых строго соблюдаются методы объектно-ориентированного программирования. В этом случае возможной является ситуация, когда работа B преобразует результат работы A, и хотя её результат будет правильным только если она и работа A выполнены без ошибок, но работу B нет необходимости переделывать, если в предшествующей работе A обнаружена ошибка. В этом случае вполне допустимо, что рациональный алгоритм восстановления будет состоять как из проверок работ, так и операций по их переделыванию.

Вариант 2 является более общим – работа B не может быть выполнена правильно, если она основана на неправильном результате предшествующей работы A и для восстановления правильности выполнения проекта её надо переделать. Пример – проект «Ремонт квартиры», в котором работа A заключается в выравнивании стены, а работа B – наклеивание обоев. Если стены не будут выровнены (ошибка при выполнении работы A), то даже если

обои будут наклеены правильно, работу В всё равно придётся переделывать после исправления ошибки (переделывания работы А). Причём на переделывание работы В уйдёт больше времени, чем было запланировано изначально - наклеенные обои придётся снять. Приведем другой пример. Проект «Оценка коммерческой недвижимости». Суммарная работа А – определение параметров оцениваемого объекта. Работа В – расчёт стоимости. Если хотя бы один из параметров оцениваемого объекта коммерческой недвижимости будет определён неправильно, то есть будет допущена ошибка в одной из работ, входящих в суммарную работу А, то расчёт стоимости будет неверным и в том случае, когда он проведен по всем правилам. После устранения допущенной ошибки расчёт должен быть проведён заново, и он займёт столько же времени, сколько занимал изначально.

Такие проекты будем называть проектами с работами, сильно связанными по результатам (ПССР). Предложенная здесь классификация проектов дополняет результаты работ [Коньшунова А. Ю., 2013; Тихоненко К. П., Меркушова Н.И., 2013].

Если предположить, что на рисунке 1.1 изображён сетевой график ПССР, то при локализации ошибки с точностью до работ {2; 3; 4}, с учётом возможности наличия ошибок в нескольких работах и при условии, что время полной проверки работы сопоставимо с временем её переделывания, целесообразно переделать каждую из этих работ. Другими словами, алгоритм восстановления правильности выполнения проекта будет состоять в переделывании всех работ, с точностью до которых локализована ошибка.

В дальнейшем будем рассматривать именно ПССР, для которых алгоритм восстановления правильности выполнения проекта состоит в переделывании всех работ, с точностью до которых локализована ошибка. Как объекты восстановления ПССР в наибольшей степени (среди других типов проектов) отличаются от технических систем. Если в нерезервированной технической системе установлены контрольные точки, то

при возникновении любой неисправности, вызвавшей отказ, немедленно снимаются показания со всех контрольных точек, что сразу даёт локализацию неисправного блока с точностью до определённого подмножества блоков системы. При этом возможностью наличия нескольких неисправных блоков можно пренебречь! Затем осуществляется поиск неисправного блока по определённому условному алгоритму, после чего неисправный блок заменяется на заведомо исправный.

Каждому варианту расстановки контрольных точек соответствует своё разбиение множества работ проекта на подмножества работ, с точностью до которых будут локализованы ошибки в работах, которое в дальнейшем будем называть вариантом локализации ошибок. Так, при наборе контрольных точек $\{П^1; П^4\}$ множество работ проекта разбивается на следующие подмножества: {1}, {2; 3; 4}, {5}. Если же будет выбран другой вариант расстановки двух контрольных точек, например $\{П^2; П^4\}$, то множество работ проекта будет разбито на следующие подмножества: {1; 2}, {3; 4}, {5}.

Каждому конкретному варианту локализации ошибок соответствует своё время восстановления при каждой конкретной ошибке.

Поясним на примере. Допустим, что плановые длительности выполнения работ в днях таковы: t_1 = 2; t_2 = 4; t_3 = 3; t_4 = 6; t_5 =1.

При наборе контрольных точек $\{П^1; П^4\}$:

- если возникнет ошибка в работе 1, то время восстановления $T_в$ составит t_1 = 2 дня;
- если возникнет ошибка хотя бы в одной из работ {2; 3; 4}, то время восстановления $T_в$ составит 13 дней = 4 дня + 3 дня + 6 дней;
- если возникнет ошибка в работе 5, то время восстановления $T_в$ составит t_5 =1 день.

Варианту расстановки контрольных точек $\{П^2; П^4\}$, который локализует возможные ошибки с точностью до подмножеств $\{1; 2\}$, $\{3; 4\}$, $\{5\}$, соответствуют значения времени восстановления в днях $\{6; 9; 1\}$.

Время восстановления проекта при возникновении ошибок является случайной величиной значения, которая при каждом конкретном наборе контрольных точек зависит от того, какие работы будут выполнены с ошибками.

Возникает вопрос: по каким критериям сравнивать различные варианты расстановки контрольных точек в проектах и какие методы в принципе можно применить для поиска оптимального варианта?

Критерии сравнения зависят от того, является проект особо ответственным или нет.

Для особо ответственных проектов, как правило, зависимость потерь от времени задержки реализации проекта является нелинейной. Эта нелинейность такова. До определённого времени задержки потери практически являются нулевыми. После этого они либо становятся равными определённой величине, либо увеличиваются в соответствии с показательной зависимостью [Лисенков В. М., 1999; Ефанов Д. В., Плеханов П. А., 2011]. Первая из указанных зависимостей характерна для ситуации, когда за срыв проекта предусмотрены фиксированные санкции.

Для обычных проектов первая из указанных зависимостей также характерна. Однако нередко можно считать, что потери линейно связаны с временем задержки реализации проекта. Примером является ситуация, когда за задержку реализации проекта начисляется пеня от плановой стоимости проекта по определённой ставке.

Как показано выше, каждому варианту расстановки контрольных точек при каждой ошибке соответствует своя длительность реализации алгоритма восстановления правильности выполнения проекта. Она зависит от того, как локализуются ошибки при данном варианте расстановки контрольных точек.

В силу этого для любого варианта расстановки контрольных точек время задержки проекта при возникновении ошибок в работах проекта является случайной величиной. Её среднее значение является подходящей целевой функцией для сравнения вариантов расстановки контрольных точек в проекте для проектов, в которых потери линейно связаны с временем задержки реализации проекта. Проблема состоит в том, что для вычисления среднего надо уметь определять вероятности ошибок в работах проекта, а проект – по определению уникальная последовательность работ. Следовательно, даже если и допустить возможность применения методов теории вероятностей, то не приходится рассчитывать на возможность статистического определения вероятностей ошибок [Jaynes E. T.,2003; Босс В., 2015]. Заметим, что это справедливо и для так называемых типовых проектов. Необходимо разработать математические модели возникновения ошибок в проектах. Это сделано в данной диссертационной работе. Причём предложенные методы существенно различны для особо ответственных и обычных проектов.

Если потери малы до определённого момента, а затем резко возрастают, то, как будет показано далее, можно предложить целевую функцию для сравнения вариантов расстановки контрольных точек, не использующую вероятности ошибок, а связывающую потери с отсутствием больших групп, неразличимых ошибок.

Изложенное позволяет сделать следующие выводы.

1. Несмотря на наличие явной аналогии между функциональной моделью технической системы и сетевым графиком проекта, с позиции диагностики проекты и технические системы являются принципиально различными объектами восстановления. Это различие особо значимо для проектов с работами, сильно связанными по результатам (ПССР), в которых работа не может считаться выполненной правильно, если неправильно выполнена хотя бы одна из работ, ей предшествующих.

2. В отличие от технических систем контрольные точки в проекте можно выбирать не только при планировании, но и в динамике – при его реализации. Каждому варианту расстановки контрольных точек при каждой ошибке соответствует своя длительность реализации алгоритма восстановления правильности выполнения проекта. В силу этого время восстановления правильности выполнения проекта при возникновении ошибок является случайной величиной значения, которая при каждом конкретном наборе контрольных точек зависит от того, какие работы будут выполнены с ошибками.

3. При выборе целевой функции для сравнения различных вариантов расстановки контрольных точек необходимо учитывать вид зависимости потерь от времени задержки реализации проекта.

4. Для особо ответственных проектов, как правило, зависимость потерь от времени задержки реализации проекта является нелинейной. Требуется предложить представительную целевую функцию, позволяющую сравнивать варианты расстановки контрольных точек и в этом сложном случае.

5. Для обычных проектов нередко можно считать, что потери линейно связаны с временем задержки реализации проекта. Тогда среднее значение времени восстановления правильности выполнения проекта является подходящей целевой функцией для сравнения вариантов расстановки контрольных точек.

Проблема состоит в том, что для вычисления среднего надо уметь определять вероятности ошибок в работах проекта, не прибегая к статистическим методам, которые нельзя применить из-за основного свойства проекта – уникальности.

Необходимо разработать математическую модель возникновения ошибок в проектах.

В следующем параграфе будут рассмотрены существующие методы построения алгоритмов диагностирования технических систем и методы выбора автоматически контролируемых параметров в этих системах. Будет оценена возможность их адаптации к решению поставленной задачи расстановки контрольных точек в проектах с учётом указанных аналогий и различий.

1.4 Методы оптимизации в задачах технической диагностики и анализ возможности их применения для проектов

Анализ литературы по технической диагностике показывает, что в наибольшей степени методы оптимизации развиты применительно к задаче построения оптимальных условных алгоритмов поиска неисправностей [Пархоменко П. П., Согомонян Е. С., 1981; Сапожников Вл. В., Сапожников В. В., 2004; Ефанов Д. В., 2014.; Гриненко А., 2001; Брускин С., Дружаев А., Марон А., Марон М., 2017]. Эти методы можно разделить на две группы – регулярные и эвристические. Регулярные методы гарантируют нахождение алгоритма, оптимального по заданному критерию. Эти методы уменьшают вычислительные сложности по сравнению с полным перебором, но, как правило, недостаточно сильно для нахождения оптимальных алгоритмов поиска неисправностей в реальных технических системах. Эвристические методы не дают гарантии построения оптимального алгоритма поиска неисправностей, но при рациональной эвристике они обеспечивают, в большинстве случаев, построение алгоритма, близкого к оптимальному, за приемлемое время расчёта даже для сложных технических систем.

Из регулярных математических методов оптимизации для построения оптимальных условных алгоритмов поиска неисправностям применяются метод динамического программирования и метод ветвей и границ [Беллман Р., 1960; Васильев Г., 1973; Хемди А., Таха Х., 2016].

Рассмотрим подробнее процедуру построения оптимального условного алгоритма поиска неисправностей в технической системе. Как было указано ранее, для ПССР, являющихся основными типами проектов, рассматриваемых в диссертации, вопрос построения алгоритма поиска неисправностей не является актуальным. Для них алгоритм восстановления правильности выполнения проекта будет состоять в переделывании всех работ, с точностью до которых локализована ошибка. Однако он интересен применительно к проектам, для которых возможной является ситуация, когда работа В преобразует результат работы А, и хотя её результат будет правильным, только если она и работа А выполнены без ошибок, но работу В нет необходимости переделывать, если в предшествующей работе А обнаружена ошибка. Такие проекты были отнесены в предыдущем параграфе к варианту 1, для них вполне допустимо, что рациональный алгоритм восстановления будет состоять как из проверок работ, так и операций по их переделыванию.

Анализ процедуры построения оптимальных алгоритмов поиска неисправностей в технических системах позволяет более ясно понять, почему регулярные методы оптимизации не применялись для нахождения оптимальных вариантов расстановки контрольных точек в этих системах.

Процедура построения оптимальных алгоритмов поиска неисправностей в технических системах методом динамического программирования разработана для систем, в которых можно пренебречь возможностью наличия кратных неисправностей.

Для построения оптимального алгоритма поиска неисправностей должны быть заданы:

1) множество возможных неисправностей $E_1 = \{e_i\}$ и их вероятности p_i $(i = 1,2, \dots n)$;

2) множество допустимых проверок $П = \{П^j\}$ и длительности их выполнения t_j $(j = 1,2, \dots, m)$;

3) глубина диагноза – перечень подмножеств неисправностей, объединённых способом устранения;

4) соответствие между неисправностями и результатами проверок.

Информации о длительностях проверок, указанных в пункте 2, достаточно, когда время каждой проверки не зависит от того, какие проверки выполнялись до неё. Для простоты изложения предположим, что неисправностей, объединённых способом устранения, нет, и поиск должен осуществляться с точностью до неисправности, вызвавшей отказ технической системы. Для применения метода динамического программирования предлагается рассматривать систему, для которой строится алгоритм поиска неисправностей, как стохастический объект управления, перемещающийся в фазовом пространстве возможных неисправностей, под действием допустимых проверок. Координатой каждого состояния в этом пространстве является текущий перечень возможных неисправностей E. В начальный момент $E = E_1$. Конечными являются состояния, для которых $\|E\| = 1$. Задача состоит в том, чтобы перевести объект управления из начального состояния в конечное за минимальное в среднем время. С точностью до обозначений в литературе по технической диагностике предлагается следующий вид уравнения Беллмана для вычисления среднего времени поиска неисправностей из промежуточного состояния [Пархоменко П. П., Согомонян Е. С., 1981]:

$$T(E) = min\left[t_j + \sum P(E \backslash \pi_k^j) * T(E \backslash \pi_k^j)\right], \qquad (1.1)$$

где $T(E)$ – минимальное среднее время поиска неисправностей, начиная из состояния E;

t_j – время проверки $П^j$;

π_k^j – один из возможных результатов проверки $П^j$ в состоянии E;

$E\backslash\pi_k^j$– состояние, которое возникнет, если при выполнении проверки π_j получен результат π_k^j;

$P(E\backslash\pi_k^j)$ – вероятность того, что при выполнении в состоянии E проверка Π^j будет иметь результат π_k^j;

$T(E\backslash\pi_k^j)$ – минимальное среднее время поиска неисправностей, начиная из состояния $E\backslash\pi_k^j$.

В уравнении (1.1) минимум берется по всем проверкам, результат которых при состоянии E не является предопределённым, а суммирование осуществляется по всем результатам π_k^j проверки Π^j, которые возможны в состоянии E.

Вычисление ведётся, начиная от состояний, в которых возможными являются две неисправности, и заканчивается расчётом минимального среднего времени неисправности для начального состояния E_1. Соответственно, когда расчёт ведётся для состояния E, значения $T(E\backslash\pi_k^j)$ уже рассчитаны ранее, поскольку

$$\left\|E\backslash\pi_k^j\right\| < \|E\|.$$

Основные вычислительные проблемы связаны со следующим.

1. Расчёт по формуле (1.1) необходимо выполнить для всех возможных фазовых состояний E, кроме конечных. Их число, в худшем случае, составит $2^n - n$.

2. В каждом состоянии E для каждого возможного результата каждой допустимой проверки необходимо определить состояние, которое возникнет, если именно этот результат будет зафиксирован.

3. Для каждого состояния E надо рассчитать вероятности возникновения всех возможных результатов всех допустимых проверок.

Реально метод динамического программирования можно применить для построения оптимального алгоритма поиска неисправностей, если в системе возможны не более чем 20 неисправностей.

Построение оптимального алгоритма поиска неисправностей упрощается, если техническая система может быть представлена функциональной моделью. В этом случае сумма в правой части уравнения (1.1) содержит всего два элемента, но это не значительно увеличивает размерность систем, для которых можно построить оптимальные алгоритмы поиска неисправностей методом динамического программирования. Невозможно применение такого метода к реальным проектам, содержащим сотни, а иногда и тысячи работ [Терехова А. Е., Верба Н. Ю., 2013].

Рассмотрим теперь процедуру построения оптимального алгоритма поиска неисправностей методом ветвей и границ. Она состоит в следующем.

В начальном состоянии возможными являются все неисправности, и допустимыми являются все проверки, входящие в таблицу неисправностей. Если выбрать в качестве проверки, выполняемой в этом состоянии, проверку Π^j, то среднее время выполнения алгоритма поиска неисправностей заведомо будет содержать слагаемое t_j. Каждому результату π_k^j этой проверки будет однозначно соответствовать определённое подмножество $E_1\&\pi_k^j$ множества возможных неисправностей E_1. Минимальное среднее время поиска неисправностей $T(E_1\&\pi_k^j)$ в $E_1\&\pi_k^j$ неизвестно, но можно найти для него $T_{\text{нг}}\ (E_1\&\pi_k^j)$ границу – оценку снизу. Тогда границу среднего времени любого алгоритма, начинающегося с π_j, можно найти по формуле [Пархоменко П. П., Согомонян Е. С., 1981]:

$$T_{\text{нг}}(\Pi^j) = t_j + \sum P(E_1\&\pi_k^j) * T_{\text{нг}}(E_1\&\pi_k^j), \qquad (1.2)$$

где $P(E_1 \& \pi_k^j)$ – вероятность результата π_k^j в начальном состоянии E_1 или, что то же, вероятность того, что после выполнении в начальном состоянии E_1 проверки $П^j$ неисправность будет локализована с точностью до подмножества неисправностей $E_1 \& \pi_k^j$.

Суммирование осуществляется по всем возможным результатам проверки $П^j$.

Для каждой допустимой проверки найдём по формуле (1.2) нижнюю границу минимального среднего времени поиска неисправности при условии, что эта проверка является первой в искомом оптимальном алгоритме. Среди найденных значений найдём минимальное. Допустим, что оно соответствует проверке с номером w. Принимаем её в качестве наиболее перспективной первой проверки искомого алгоритма. Она может иметь определённые результаты, каждому из которых будет соответствовать определённое подмножество множества возможных неисправностей. Для каждого из них надо найти наиболее перспективную проверку, после чего вычислить по формуле, аналогичной (1.2), общую оценку минимального среднего времени поиска неисправностей по алгоритму, в котором найденные проверки будут следовать за $П^w$. Если эта оценка меньше, чем любая другая из найденных на предыдущем шаге, то продолжаем достраивать алгоритм в этом направлении. Если среди ранее найденных есть меньшая граница, то достраиваем алгоритм, начинающийся с проверки, ей соответствующей. Продолжаем аналогичный процесс, пока не будет построен алгоритм, различающий неисправности с требуемой глубиной диагноза.

Эффективность метода ветвей и границ зависит от того, насколько удачно определена граница. По длительности вычислений нередко он хуже полного перебора. Причём, в отличие от метода динамического программирования, его вычислительную сложность невозможно предсказать заранее.

Невозможность использовать регулярные методы для построения алгоритмов поиска неисправностей в реальных сложных технических системах привела к развитию эвристических подходов к построению таких алгоритмов. Наиболее логически обоснованным из них явился информационный подход Джонсона, впервые приведенный им в работе [Jonson R., 1960]. Р. Джонсон исходил из следующего. Поиск неисправности – это последовательность экспериментов-проверок. Каждый эксперимент уменьшает неопределённость того, какая неисправность технической системы произошла. Следуя понятиям теории информации Клода Шеннона [Shannon C. E., 1948; Шеннон К.; 1963; Духин А. А., 2007; Кудряшов Б. Д., 2010; Чернавский Д., 2016], можно сказать, что каждая допустимая проверка даёт информацию о неисправности, или, другими словами, существует средняя взаимная информация случайных величин N – номер неисправности и π^j – результат проверки $П^j$. Количество этой информации по Шеннону определяется по формуле:

$$I(N;\ \pi^j)\ =\ H(N)\ -\ H(N/\pi^j), \qquad (1.3)$$

где $H(N)$ – исходная энтропия случайной величины N;

$H(N/\pi^j)$ –- остаточная энтропия.

Чем больше информации даёт проверка, тем лучше. С другой стороны, её выполнение требует определённых затрат времени. Чем они больше, тем хуже. Р. Джонсон предложил в каждом возможном состоянии объекта поиска неисправностей, которое может возникнуть в результате выполнения уже внесённых в алгоритм проверок, добавлять в него ту проверку, для которой максимальным является отношение количества информации, которое она приносит, к времени её выполнения.

Так, в начальном состоянии E_1 согласно критерию Джонсона, надо выполнять такую проверку $П^*(E_1)$, которой соответствует максимум отношения информации к длительности:

$$\Pi * (E_1) \rightarrow max\,[I(N;\,\pi^j)/t_j] \qquad (1.4)$$

учитывая, что максимум берётся по всем j.

После того как Π^1 принята в качестве первой проверки алгоритма поиска неисправностей, необходимо определить состояния, которые возникнут при её различных результатах, и для каждого из них снова применить критерий Джонсона для выбора проверок, которые надо включить в алгоритм. При этом для каждого состояния при вычислении информации необходимо пересчитать вероятности неисправностей, используя формулу Байеса.

Главным недостатком критерия Джонсона явилась несоразмерность величин, стоящих в числителе и знаменателе. Шкала изменения информации является логарифмической. Шкала изменения времени – линейная. В результате небольшие различия во времени выполнения сравниваемых проверок оказываются доминирующим фактором, даже при их существенно различной информативности. В силу этого критерий Джонсона является неустойчивым к неизбежным на практике малым ошибкам в определении времени проверок. Вместе с тем, когда время выполнения одинаково для любой проверки, информационный критерий Джонсона позволяет находить алгоритмы поиска неисправностей, среднее время реализации которых близко к минимальному. Заметим, что если ещё каждая проверка может иметь только два результата, то в этом частном случае процедура, предписываемая критерием Джонсона для построения алгоритма поиска неисправностей в технической системе, становится аналогичной методу Шеннона – Фано, предложенному для построения оптимальных неравномерных кодов. Заметим, что метод Шеннона – Фано также является не регулярным, а эвристическим.

Проанализируем теперь методы оптимизации выбора контрольных точек в технических системах на предмет возможности их применения для решения поставленной проблемы выбора контрольных точек в проектах.

Две задачи о контрольных точках сформулированы в литературе по технической диагностике как строго оптимизационные: задача об определении минимального набора контрольных точек, достаточных для контроля исправности; и задача об определении минимального набора контрольных точек, достаточных для поиска неисправности с требуемой глубиной. Они решались как для технических объектов, представленных функциональной моделью, так и для общего случая, когда соответствие между результатами допустимых проверок и возможными неисправностями представлено в виде не двоичной таблицы неисправностей. Не во всех работах, посвящённых данной тематике, речь шла о контрольных точках [Марон В., 2011]. В ряде работ аналогичные задачи решались применительно к наборам проверок, время выполнения которых одинаково. В некоторых работах вместо термина «контрольные точки» использовался термин «контролируемые параметры».

Первая из этих задач полностью решена для технических систем, которые можно представить в виде функциональной модели. Доказано, что для контроля исправности необходимо и достаточно установить контрольные точки на всех выходных вершинах функциональной модели или, что то же, выходных вершинах графа, соответствующего функциональной модели объекта диагноза. Тогда факт возникновения неисправности в любом блоке системы будет обнаруживаться [Карибский В. В., Пархоменко П. П., Согомонян Е. С., Халчев В. Ф., 1976].

В общем случае задача о минимальном наборе контрольных точек, достаточном для контроля исправности, сводится к задаче нахождения минимального набора столбцов таблицы неисправностей, в котором для любой строки есть результат одной из проверок, отличающийся от того,

который эта проверка имеет при исправном состоянии системы. Для её решения предложен ряд эвристических подходов.

Задача определения минимального набора контрольных точек, достаточных для поиска неисправностей с требуемой глубиной, также сводится в общем случае к поиску минимального набора столбцов таблицы неисправностей, обладающего определёнными свойствами. Так, если неисправностей, объединённых способом устранения, нет, и требуется однозначно различать все неисправности, то это набор столбцов, при котором каждой строке соответствует свой элемент, не совпадающий ни с одним другим хотя бы в одном столбце. Для определения такого набора предложены эвристические методы.

Минимальный набор контрольных точек, достаточный для контроля исправности технической системы, как правило, обеспечивает недостаточную локализацию неисправностей, в результате чего длительность последующего поиска неисправностей может быть недопустимо велика и приводить к значительным потерям при отказе этой системы.

Приведём простой, но наглядный пример. Функциональная модель системы представляет собой последовательную цепочку из n блоков. Для контроля исправности достаточно установить контрольную точку на выходе последнего блока – блока с номером n. При этом исправность всех n блоков контролируется, но их неисправности неразличимы! При любой из них в контрольной точке будет зафиксирован отрицательный результат. Для локализации же неисправностей такой системы с точностью до неисправного блока потребуется установить ещё $(n-1)$ контрольную точку (по контрольной точке на выходе каждого промежуточного блока) в дополнение к контрольной точке, установленной для контроля исправности системы. Таким образом, n контрольных точек необходимы для контроля исправности и поиска неисправностей в простейшей системе, представленной функциональной моделью из n строго последовательных блоков! Набор

контрольных точек, позволяющий контролировать исправность системы и локализовать все возможные неисправности с требуемой глубиной, уместно назвать полным набором контрольных точек.

Показания в контрольных точках технических систем снимаются с помощью датчиков. Для полного набора контрольных точек требуемое количество датчиков, как правило, слишком велико, чтобы их установка была оправдана с финансовой точки зрения. Поэтому возникает вопрос о построении набора контрольных точек, не являющегося полным, но в то же время уменьшающего время восстановления технической системы. Назовём такой набор дополнительным набором контрольных точек. Здесь просматривается аналогия с проектами, где стратегия полной проверки каждой работы нецелесообразна, и необходимо выбрать только отдельные работы для полной проверки. Наиболее проработанной в технической диагностике является проблема построения тестов для схем и дискретных устройств. Эта проблема рассматривается на сегодняшний день как основная. При генерации набора тестов в первую очередь важно достичь требуемой глубины диагноза, длительность тестирования, учитывая высокую скорость каждого отдельного теста, менее критична. Здесь созданы как технологии генерации наборов тестов для поиска неисправностей [Сапожников Вл. В., Сапожников В. В., 2004], так и методы моделирования неисправностей и даже сбоев [Петросянц К. О., Самбурский Л. М., Харитонов И. А., 2017].

Проблема построения оптимального набора контрольных точек для поиска (локализации) неисправностей, которая относится в первую очередь к непрерывным системам, проработана значительно в меньшей степени, чем проблема тестирования дискретных систем. Так, в литературе по технической диагностике отсутствует математическая постановка задачи выбора оптимального дополнительного набора контрольных точек. Соответственно не идёт речь и о применении регулярных методов оптимизации. В книге [Дмитренко И. Е., Сапожников В. В., Дьяков Д. В.,

1994] изложен эвристический метод определения дополнительного набора контрольных точек, основанный на идее Джонсона. Оценка его эффективности не проводилась, поскольку целевая функция для сравнения вариантов не была определена. При этом число точек считается выбранным заранее. Кроме того, делается предположение об отсутствии кратных неисправностей. Оно полностью обосновано для технических систем, в которых:

1) дефекты в элементах возникают независимо друг от друга – отсутствуют элементы, дефекты которых приводят к дефектам других элементов;

2) нет резервированных элементов.

Вероятности неисправностей в элементах рассчитываются на основании характеристик их безотказности. В случае непрерывных элементов в качестве такого показателя принимается интенсивность отказов, а для дискретных, например, реле, среднее количество срабатываний до отказа [Гнеденко Б., Беляев Ю., Соловьёв А., 2013; Каштанов В. А., Медведев А. И., 2010].

Сопоставив это с приведенными в предыдущем параграфе особенностями проектов как объектов диагноза, можно отметить следующее.

Предположение о наличии единственной неисправности, основанное, в числе прочего, на предположении о независимых дефектах, делает методы, предложенные в работах по определению так называемого рационального дополнительного набора контрольных точек, не применимыми к проектам с работами, сильно связанными по результатам (ПССР), которые являются основным типом проектов, рассматриваемых в данной диссертации.

Принятые методы расчёта вероятностей неисправностей элементов технических систем невозможно использовать для расчёта вероятностей ошибок в работах проекта.

Изложенное позволяет сделать следующие выводы.

1. В наибольшей степени методы оптимизации развиты в технической диагностике применительно к задаче построения оптимальных условных алгоритмов поиска неисправностей. Из регулярных математических методов оптимизации для построения оптимальных условных алгоритмов поиска неисправностей применяются метод динамического программирования и метод ветвей и границ. При этом метод ветвей и границ применяется, только когда целевой функцией является среднее время поиска неисправности. Метод динамического программирования применялся изначально также только для построения условного алгоритма поиска неисправностей оптимального по критерию минимума среднего времени поиска. Позднее было показано, что его можно применять с определёнными изменениями и тогда, когда потери при отказе системы не линейны, а возрастают экспоненциально с ростом времени восстановления.

2. При современном уровне развития вычислительной техники регулярные методы не могут гарантировать построение оптимального алгоритма поиска неисправностей для технических систем, в которых число возможных неисправностей больше, чем 25–30. Невозможность использовать регулярные методы для построения алгоритмов поиска неисправностей в реальных сложных технических системах привела к развитию эвристических подходов к построению таких алгоритмов. Наиболее логически обоснованным из них явился информационный подход Джонсона, получивший название «критерий Джонсона». Главным недостатком критерия Джонсона явилась несоразмерность величин, стоящих в числителе и знаменателе. Шкала изменения информации является логарифмической. Шкала изменения времени – линейная. В результате небольшие различия во времени выполнения сравниваемых проверок оказываются доминирующим фактором даже при их существенно различной информативности. В силу

этого критерий Джонсона является неустойчивым к неизбежным на практике малым ошибкам в определении времени проверок.

3. Две задачи о контрольных точках сформулированы в литературе по технической диагностике как строго оптимизационные: задача об определении минимального набора контрольных точек, достаточных для контроля исправности; и задача об определении минимального набора контрольных точек, достаточных для поиска неисправности с требуемой глубиной. Первая из них полностью решена для технических систем, которые можно представить функциональной моделью. Для решения второй из них предложены эвристические методы. Набор контрольных точек, позволяющий контролировать исправность системы и локализовать все возможные неисправности с требуемой глубиной, уместно назвать полным набором контрольных точек. Как правило, он включает в себя слишком много элементов, чтобы его было целесообразно реализовать на практике. Поэтому возникает вопрос о построении набора контрольных точек, не являющегося полным, но в тоже время уменьшающего время восстановления технической системы. Здесь просматривается аналогия с проектами, где стратегия полной проверки каждой работы нецелесообразна, и необходимо выбрать только отдельные работы для полной проверки.

4. Математическая постановка задачи построения оптимального дополнительного набора контрольных точек в литературе по технической диагностике отсутствует. Не выбрана целевая функция для сравнения вариантов. Соответственно не идёт речь и о применении регулярных методов оптимизации. Вместе с тем имеются работы, в которых для построения дополнительного набора контрольных точек применён эвристический подход, подобный критерию Джонсона. Оценка его эффективности не приведена.

5. Учитывая особенности проектов как объектов диагноза, приведенные в предыдущем параграфе, можно отметить следующее.

Предположение о наличии единственной неисправности, основанное, в числе прочего, на предположении о независимых дефектах, делает методы, предложенные в работах по определению так называемого рационального дополнительного набора контрольных точек, не применимыми к проектам с работами, сильно связанными по результатам (ПССР), которые являются основным типом проектов, рассматриваемых в данной диссертации. Кроме того, принятые методы расчёта вероятностей неисправностей элементов технических систем невозможно использовать для расчёта вероятностей ошибок в работах проекта.

1.5 Выводы по главе 1

Проведенный анализ позволяет сделать следующие выводы.

1. При выполнении любой работы проекта может возникнуть ошибка, то есть отклонение от установленных требований. Если ошибки в работах будет выявлена только на этапе проверки соответствия конечного продукта установленным требованиям, то потребуется много времени для их локализации. Сроки проекта могут оказаться сорванными.

2. Целесообразной является стратегия своевременного обнаружения ошибок в работах проекта, при которой каждая из работ проверяется на соответствие важнейшим параметрам (частичная проверка), а полные проверки осуществляются после некоторых работ – контрольных точек, выбираемых при планировании проекта.

3. Возникновение ошибок при выполнении работ проекта следует классифицировать как отрицательный риск. Локализация этих ошибок с помощью контрольных точек является методом снижения влияния риска на результаты проекта.

4. Для каждого крупного проекта существует огромное количество возможных наборов контрольных точек. Рациональный выбор контрольных точек является одной из важнейших задач диагностики проектов.

5. Работы, посвящённые управлению проектами, не дают ответа на вопрос о том, как и в каком количестве выбирать контрольные точки в проектах. Под диагностикой проектов в существующей литературе по управлению проектами понимается в первую очередь определение показателей, позволяющих установить, что произошло отклонение от базового плана, требующее перепланирования или принятия других управленческих решений.

6. Вместе с тем имеются чёткие рекомендации по снижению вероятности возникновения ошибок в работах для особо ответственных проектов. Это пооперационный контроль, а для проектов разработки программных продуктов – методология объектно-ориентированного программирования, которая существенно упрощает локализацию ошибок с помощью контрольных точек.

7. Наиболее полно вопросы проверки правильности функционирования технических и программных систем, а также вопросы локализации неисправностей решаются в науке, названной «техническая диагностика».

8. Аналогия между сетевой моделью проекта и функциональной моделью технической системы, построенной по блочному принципу, предполагает потенциальную возможность применить ряд методов, разработанных в технической диагностике для решения задач диагностики проектов.

9. Проект диагностируется в процессе реализации. Это главное отличие проектов от технических систем при постановке и решении задач диагностики. Поэтому прямой перенос результатов, полученных для технических систем, невозможен.

10. Несмотря на наличие явной аналогии между функциональной моделью технической системы и сетевым графиком проекта, с позиции диагностики проекты и технические системы являются принципиально различными объектами восстановления. Это различие особо значимо для

проектов с работами, сильно связанными по результатам (ПССР), в которых работа не может считаться выполненной правильно, если неправильно выполнена хотя бы одна из работ, ей предшествующих.

11. В отличие от технических систем контрольные точки в проекте можно выбирать не только при планировании, но и в динамике – при его реализации.

12. Каждому варианту расстановки контрольных точек при каждой ошибке соответствует своя длительность восстановления правильности выполнения проекта. В силу этого время восстановления правильности выполнения проекта при возникновении ошибок является случайной величиной, значения которой при каждом конкретном наборе контрольных точек зависит от того, какие работы будут выполнены с ошибками.

13. При выборе целевой функции для сравнения различных вариантов расстановки контрольных точек необходимо учитывать вид зависимости потерь от времени задержки реализации проекта.

14. Для особо ответственных проектов, как правило, зависимость потерь от времени задержки реализации проекта является нелинейной. Требуется предложить представительную целевую функцию, позволяющую сравнивать варианты расстановки контрольных точек и в этом сложном случае.

15. Для обычных проектов нередко можно считать, что потери линейно связаны с временем задержки реализации проекта. Тогда среднее значение времени восстановления правильности выполнения проекта является подходящей целевой функцией для сравнения вариантов расстановки контрольных точек.

16. Проблема состоит в том, что для вычисления среднего надо уметь определять вероятности ошибок в работах проекта, не прибегая к статистическим методам, которые нельзя применить из-за основного

свойства проекта – уникальности. Необходимо разработать математическую модель возникновения ошибок в проектах.

17. В наибольшей степени методы оптимизации развиты в технической диагностике применительно к задаче построения оптимальных условных алгоритмов поиска неисправностей. Из регулярных математических методов оптимизации для построения оптимальных условных алгоритмов поиска неисправностей применяются метод динамического программирования и метод ветвей и границ. При этом метод ветвей и границ применяется, только когда целевой функцией является среднее время поиска неисправности. Метод динамического программирования применялся изначально также только для построения условного алгоритма поиска неисправностей оптимального по критерию минимума среднего времени поиска. Позднее было показано, что его можно применять с определёнными изменениями и тогда, когда потери при отказе системы не линейны, а возрастают экспоненциально с ростом времени восстановления.

18. При современном уровне развития вычислительной техники регулярные методы не могут гарантировать построение оптимального алгоритма поиска неисправностей для технических систем, в которых число возможных неисправностей больше, чем 25–30. Невозможность использовать регулярные методы для построения алгоритмов поиска неисправностей в реальных сложных технических системах привела к развитию эвристических подходов к построению таких алгоритмов. Наиболее логически обоснованным из них явился информационный подход Джонсона, получивший название «критерий Джонсона». Главным недостатком критерия Джонсона явилась несоразмерность величин, стоящих в числителе и знаменателе. Шкала изменения информации является логарифмической. Шкала изменения времени – линейная. В результате небольшие различия во времени выполнения сравниваемых проверок оказываются доминирующим фактором даже при их существенно различной информативности. В силу

этого критерий Джонсона является неустойчивым к неизбежным на практике малым ошибкам в определении времени проверок.

19. Две задачи о контрольных точках сформулированы в литературе по технической диагностике как строго оптимизационные: задача об определении минимального набора контрольных точек, достаточных для контроля исправности; и задача об определении минимального набора контрольных точек, достаточных для поиска неисправности с требуемой глубиной. Первая из них полностью решена для технических систем, которые можно представить функциональной моделью. Для решения второй из них предложены эвристические методы. Набор контрольных точек, позволяющий контролировать исправность системы и локализовать все возможные неисправности с требуемой глубиной, уместно назвать полным набором контрольных точек. Как правило, он включает в себя слишком много элементов, чтобы его было целесообразно реализовать на практике. Поэтому возникает вопрос о построении набора контрольных точек, не являющегося полным, но в то же время уменьшающего время восстановления технической системы. Здесь просматривается аналогия с проектами, где стратегия полной проверки каждой работы нецелесообразна, и необходимо выбрать только отдельные работы для полной проверки.

20. Математическая постановка задачи построения оптимального дополнительного набора контрольных точек в литературе по технической диагностике отсутствует. Не выбрана целевая функция для сравнения вариантов. Соответственно не идёт речь и о применении регулярных методов оптимизации. Вместе с тем имеются работы, в которых для построения дополнительного набора контрольных точек применён эвристический подход, подобный критерию Джонсона. Оценка его эффективности не приведена.

21. Учитывая особенности проектов как объектов диагноза, приведенные в предыдущем параграфе, можно отметить следующее.

Предположение о наличии единственной неисправности, основанное, в числе прочего, на предположении о независимых дефектах, делает методы, предложенные в работах по определению так называемого рационального дополнительного набора контрольных точек, не применимыми к проектам с работами, сильно связанными по результатам (ПССР), которые являются основным типом проектов, рассматриваемых в данной диссертации. Кроме того, принятые методы расчёта вероятностей неисправностей элементов технических систем невозможно использовать для расчёта вероятностей ошибок в работах проекта.

На основании этого поставлены следующие задачи диссертационного исследования.

1. Разработать диагностическую модель проекта. Эта модель должна давать возможность установить связь между ошибками в работах и результатами проверок.

2. Разработать математическую модель возникновения ошибок в работах, которая позволит рассчитывать вероятности ошибок.

3. Предложить критерии для сравнения вариантов расстановки контрольных точек в зависимости того, как потери от задержки окончания проекта, вызванной ошибками в работах, зависят от времени восстановления правильности его выполнения.

4. Разработать методы определения наборов контрольных точек в соответствии с предложенными критериями и экспериментально оценить их эффективность, если это будут эвристические, а не регулярные методы, гарантирующие нахождение оптимума.

5. Разработать алгоритмы и программное обеспечение для экспериментальной проверки эффективности предложенных методов и их практического применения к расстановке контрольных точек в реальных проектах.

Решению этих задач посвящены главы 2 и 3 диссертации.

Глава 2 Разработка методов расстановки контрольных точек

2.1 Диагностическая модель проекта

Перейдём непосредственно к разработке диагностической модели проекта, предназначенной для решения задач контроля правильности выполнения проекта и поиска возможных ошибок. При этом ограничимся рассмотрением проектов с работами, сильно связанными по результатам (ПССР). Напомним, что, как указано в параграфе 1.3, это проекты, в которых работа не может быть выполнена правильно, если она основана на неправильном результате предшествующей работы.

Определение. Диагностическая модель проекта (ДМП) – это форма представления соответствия между ошибками в работах проекта и результатами полных проверок работ, при которой проект представляется графом сетевой модели, где вершины графа представляют работы, а дуги – не только логические связи между работами, но и места выполнения возможных полных проверок. При этом полная проверка работы j имеет положительный результат тогда и только, когда ошибки отсутствуют в данной работе и всех работах, ей предшествующих. В противном случае эта проверка имеет отрицательный результат.

Положительный результат полной проверки работы j – это результат, при котором установлено, что все параметры результата этой работы соответствуют установленным требованиям. Отрицательный результат полной проверки работы j показывает, что по крайней мере один из параметров результата этой работы не соответствует установленным требованиям. Причиной отрицательного результата полной проверки работы j может являться ошибка при выполнении этой работы или ошибки в

одной из работ, ей предшествующих. При этом под предшествующими понимаются все работы, лежащие на путях от начальных работ – начальных вершин графа модели к данной.

Применение предложенной ДМП позволяет построить таблицу возможных ошибок (ТВО), в которой для каждой полной проверки будет указан её результат при каждой возможной ошибке, используя методы и алгоритмы теории графов [Алескеров Ф. Т., Хабина Э. Л., Шварц Д. А., 2012]. По аналогии с таблицей возможных неисправностей технической системы такую таблицу будем называть таблицей возможных ошибок (ТВО) проекта. Если в проекте n работ, то это квадратная таблица размерности $n \times n$.

Укажем, как её построить. Обозначим положительный результат полной проверки символом «0» – ошибок нет, а отрицательный символом «1» – ошибка есть. Зададим матрицу смежности ДМП. Это матрица размерности $n \times n$, элемент $(i; j)$ которой «1», если на ДМП есть дуга из вершины i в вершину j, в противном случае этот элемент «0». Элементы главной диагонали матрицы смежности ДМП всегда равны нулю, поскольку на сетевой модели проекта циклов быть не может. Построим матрицу путей (матрицу связности) ДМП. Это матрица размерности $n \times n$, элемент $(i; j)$ которой «1», если на ДМП есть путь из вершины i в вершину j, в противном случае этот элемент «0». Для построения можно использовать любой из методов, разработанных в теории графов. Заменим в матрице путей ДМП все элементы главной диагонали на «1». Полученная таблица есть ТВО проекта. Элемент $(i; j)$ этой таблицы «1», если при ошибке в работе, имеющей номер i в ДМП, полная проверка работы с номером j будет иметь отрицательный результат (ошибка обнаруживается); если же при ошибке в работе, имеющей номер i в ДМП, полная проверка работы с номером j всё равно будет иметь

положительный результат (ошибка не обнаруживается), то элемент $(i; j)$ имеет значение «0».

Используя ДМП, можно легко решить задачу определения минимального полного набора проверок, необходимого и достаточного для контроля правильности выполнения проекта. По аналогии с результатом, полученным для технических систем, представленных функциональной моделью, можно показать, что такой набор включает в себя проверки конечных работ проекта.

Конечные работы элементарно находятся по матрице смежности ДМП – им соответствуют нулевые строки.

В дальнейшем для простоты изложения будем предполагать, что в проекте одна конечная работа. Заметим, что если в реальности в проекте несколько конечных работ, то всегда можно создать на сетевом графике веху, для которой они будут предшественниками.

Задача определения минимального набора проверок, достаточного для поиска ошибки с точностью до работы, также может быть решена с использованием ДМП, хотя её решение несколько сложнее. Не будем здесь на этом останавливаться, поскольку, как правило, такой набор состоит из слишком большого числа элементов, чтобы его можно было принять в качестве набора контрольных точек (см. параграф 1.4).

Изложенное позволяет сделать следующие выводы.

1. В настоящем параграфе впервые предложена диагностическая модель проекта (ДМП).

2. Она позволяет представить соответствие между результатами проверок и возможными ошибками в работах в виде таблицы возможных ошибок (ТВО) проекта, для построения которой можно использовать методы, разработанные в теории графов.

3. Используя ДМП, можно легко решить задачу определения минимального полного набора проверок, необходимого и достаточного для контроля правильности выполнения проекта. Она является основой для решения поставленной задачи определения наборов контрольных точек в проектах.

2.2 Математическая модель возникновения ошибок в работах проекта

Для решения задачи выбора контрольных точек в проекте необходимо знать вероятности ошибок в работах. Однако по определению каждый проект уникален, даже если он является типовым. В противном случае это уже не проект, а повторяемый бизнес-процесс [Громов А. И., Фляйшман А., Шмидт В., 2016; Хаммер М., Чампи Дж., 1999; Черемных О. С., Черемных С. В., 2005; Томорадзе И., Дмитрик А., 2013].

В силу этого определить искомые вероятности ошибок статистическими методами, как правило, невозможно. Даже если строительная компания осуществляет проекты возведения типовых жилых домов и для каждого реализованного проекта фиксирует, при выполнении каких работ возникали ошибки, то всё равно этих данных недостаточно для правдоподобного определения искомых вероятностей. Причина здесь даже не в том, что частота сходится к вероятности достаточно медленно, важнее, что ошибки могут возникать в нескольких работах проекта. Для $\boldsymbol{n}$ работ возможными являются $\boldsymbol{n}!$ вариантов. Собрать статистические данные для определения вероятности появления каждого из этих вариантов невозможно. В силу этого актуальной является проблема разработки математической модели возникновения ошибок в работах проекта. Такая модель разработана автором [Maron M. A., 2018].

Предлагается построить математическую модель процесса возникновения ошибок в проекте на базе пуассоновского потока событий. Для этого процесс возникновения ошибок должен физически обладать свойствами: ординарности, отсутствия последействия и стационарности [Вентцель Е. С., 1972; Каштанов В. А., Ивченко Г. И., Коваленко И. Н., 2012; Кирпичников А., 2018; Демидович Б., Марон И., 2006; Шишкин Е., 2008]. Начнём с ординарности. Она заключается в том, что ошибки должны возникать как одиночные, а не групповые события. В большинстве случаев это свойство выполняется. Даже если работы по графику производятся параллельно, то ошибки каждый исполнитель допускает самостоятельно. Ситуацией, когда, например, при строительстве может нарушиться ординарность, является землетрясение, но это катастрофическое явление. Отсутствие последействия состоит в том, что вероятность ошибки в очередной работе не должна зависеть от того, сколько ошибок уже возникло в ранее выполненных работах. Это свойство будет соблюдаться, если разные работы выполняют разные исполнители. Если же исполнитель один и никаких мер воздействий к нему в связи с допущенными ошибками не применяется, то свойство отсутствия последействия не будет выполнено. Стационарность состоит в том, что вероятность возникновения ошибки должна зависеть только от длительности работы и не должна зависеть от того, как расположена работа на графике проекта. Независимость от расположения на графике может нарушаться, если по мере приближения к сроку окончания проекта начинается аврал. Если же налажен бескомпромиссный контроль за ходом выполнения работ, а именно так должно быть, то можно считать, что независимость вероятности ошибки от её расположения на графике проекта соблюдается.

Можно утверждать, что если при реализации проекта не допускается ослабления контроля качества выполнения работ даже по мере приближения запланированных сроков его окончания, к недобросовестным исполнителям

своевременно применяются меры воздействия, заставляющие их соблюдать установленные требования; то поток ошибок в работах проекта можно считать ординарным, с отсутствием последействия и обладающим свойством индифферентности к расположению работы на временном графике проекта. Заметим, что требования к организации и контролю работ, обеспечивающие эти свойства, указаны как обязательные не только в PMBoK, но и практически во всех стандартах управления проектами. Сложнее обстоит дело с тем, чтобы выполнялось первое условие стационарности – зависимость вероятности возникновения ошибки только от длительности работы. Оно заведомо выполняется, если все работы осуществляют квалифицированные специалисты, сопоставимыми по своим навыкам в порученных работах и с применением материалов сопоставимого качества. Тогда каждая ошибка – результат случайности. Если же заранее известно, что одна бригада лучше другой, то будет уместно предположить, что вероятность ошибки зависит не только от длительности работы. Заметим, что если серьёзных оснований для таких суждений нет, то следует считать, что стационарность выполняется. Предлагаемая ниже модель применима к проектам, при реализации которых выполнены указанные выше требования, обеспечивающие ординарность, отсутствие последействия и стационарность потока ошибок.

Математическая модель процесса возникновения ошибок в работах проекта должна отвечать следующим требованиям.

1. Соответствовать реальностям процесса возникновения ошибок.

2. Быть удобной для проведения расчётов, связанных с диагностикой и анализом мер по снижению негативных последствий ошибок в работах проекта.

3. Параметры модели можно определить на основании реально доступных статистических данных о реализации аналогичных проектов.

Непреложным требованием является полное соответствие модели логике того раздела математики, на базе которого она создана. Предлагается следующая математическая модель процесса возникновения ошибок в работах проекта.

Имеется проект, состоящий из n работ. Длительность работы i равна t_i $(i = 1,2, \dots n)$. На выполняемые работы действует простейший поток ошибок. Интервал действия равен τ – суммарному времени выполнения всех работ – сумме t_i по i от 1 до n. Заметим, что τ отличается от времени выполнения проекта, поскольку работы могут выполняться параллельно. Тогда вероятность P_k того, что ровно k работ будут выполнены с ошибками, можно найти по формуле Пуассона:

$$P_k = \frac{a^k}{k!} e^{-a}, \qquad (2.1)$$

где a – среднее число ошибок за время τ.

Величина a и интенсивность потока ошибок λ (среднее число ошибок в единицу времени) связаны соотношением:

$$a = \lambda \cdot \tau. \qquad (2.2)$$

Если проект типовой, то вполне реально считать, что известен процент аналогичных проектов, выполненных без ошибок. Или, что то же, известна P_0 – вероятность того, что ошибок в проекте не будет. В соответствии с формулами (2.1) и (2.2) имеем:

$$P_0 = e^{-\lambda\tau}, \qquad (2.3)$$

откуда:

$$\lambda = -\frac{lnP_0}{\tau}. \quad (2.4)$$

Естественно, формула (2.4) не может применяться, если $P_0 = 0$. В этом случае интенсивность потока отказов придётся определять, основываясь на формуле Пуассона (2.1) при $k \neq 0$.

При простейшем потоке ошибок вероятность того, что работа i будет выполнена с ошибкой, можно найти по формуле:

$$p_i = p(t_i) = 1 - e^{-\lambda t_i}. \quad (2.5)$$

Соответственно, вероятность того, что работа будет выполнена правильно, без ошибки, можно найти по формуле:

$$q_i = q(t_i) = e^{-\lambda t_i}. \quad (2.6)$$

Нетрудно убедиться, что в соответствии с формулой (2.5), вероятность ошибки убывает при уменьшении длительности работы и в пределе стремится к нулю. С помощью предложенной модели можно найти вероятность любой из возможных комбинаций ошибок в работах. Также можно проверить, что вероятность возникновения хотя бы одной ошибки дополняет до единицы P_0 – вероятность того, что ошибок в проекте не будет. Таким образом, предложенная модель для расчёта вероятностей ошибок в работах проекта полностью соответствует логике теории случайных процессов [Каштанов В. А., Ивченко Г. И., Коваленко И. Н. 2012; Каштанов В. А., Медведев А. И., 2010], на базе которого она создана.

Предложенная модель соответствует реальностям процесса возникновения ошибок, если при реализации проекта не допускается ослабления контроля качества выполнения работ даже по мере приближения

запланированных сроков его окончания, к недобросовестным исполнителям своевременно применяются меры воздействия, заставляющие их соблюдать установленные требования; и нет оснований предполагать, что одни исполнители работают лучше других. Для определения среднего числа проектов, выполненных без ошибок, требуется намного меньше статистических данных, чем для определения вероятностей. Соответственно, параметры модели можно определить на основании реально доступных статистических данных о реализации аналогичных проектов.

Предложенная модель возникновения ошибок в работах проекта является общей и применима для широкого круга реальных проектов. Для решения задачи определения набора контрольных точек в особо ответственных проектах, при реализации которых для всех работ выполняется пооперационный контроль, можно предложить другую модель расчёта вероятностей ошибок. Такие проекты будем в дальнейшем называть ООП.

Пусть имеется ООП, состоящий из n работ. Время выполнения работы j равно t_j. Допустим, что конечный контроль показал факт того, что проект выполнен неправильно. Как отмечено в параграфе 1.1, для ООП, решая задачу выбора контрольных точек, в расчётах можно пренебречь малыми вероятностями кратных ошибок. Требуется определить распределение вероятностей ошибок по работам проекта при условии, что ошибка есть.

Для определения искомых условных вероятностей p_i $(i = 1,2,\dots,n)$. будем исходить из следующего.

1. Длительность работы отражает её сложность. Соответственно, чем больше длительность работы, тем больше вероятность возникновения ошибки при её выполнении.

2. Ошибка есть (наличие ошибки – достоверное событие) и только одна. Следовательно, сумма всех p_i равна 1.

Эти условия будут удовлетворены, если принять, что:

$$p_i = \frac{t_i}{\sum_{j=1}^{n} t_j} = \frac{t_i}{\tau}. \qquad (2.7)$$

Приведём пример. Допустим, ДМП состоит из 8 последовательных работ. Время каждой работы в днях равно её номеру: t_1 = 1 день, t_2 = 2 дня, …, t_8 = 8 дней. Проект является типовым. Известно, что 60% подобных проектов были выполнены правильно – без ошибок. Тогда, $P_0 = 0{,}6$; $\tau = 36$ дней, и, в соответствии с формулой (2.3), интенсивность потока ошибок $\lambda = 0{,}014$ [1/день]. Вероятность возникновения ошибки в любой из работ можно найти по формуле (2.5). Так, вероятность возникновения ошибки в работе 4 будет равна:

$$p_4 = 1 - e^{-(4*0{,}014)} = 0{,}055.$$

Заметим, что в данном расчёте не предполагалось ни то, что ошибки в проекте есть, ни то, эти ошибки не являются кратными!

Допустим теперь, что рассматривается ООП и надо определить вероятность ошибки в работе 4 при условии, что при выполнении проекта допущена ошибка в одной и только одной работе, тогда, в соответствии с формулой (2.7), получим $p_4 = 4/36 = 0{,}111$.

Это вероятность наличия ошибки в работе 4 при условии, что при выполнении проекта допущена одна и только одна ошибка!

Изложенное позволяет сделать следующие выводы.

1. Впервые разработана математическая модель возникновения ошибок в работах проекта. Модель соответствует реальностям процесса возникновения ошибок, если: ошибки возникают независимо друг от друга; при реализации проекта не допускается ослабления контроля качества выполнения работ даже по мере приближения запланированных сроков его окончания; к недобросовестным исполнителям своевременно применяются

меры воздействия, заставляющие их соблюдать установленные требования; нет оснований предполагать, что одни исполнители работают лучше других.

2. С помощью предложенной модели можно рассчитать вероятности не только для одиночных ошибок, но и для любой их комбинации.

3. Предложена и модель расчёта вероятностей ошибок для ООП. В этом частном, но исключительно важно случае, выбор варианта расстановки контрольных точек уместно проводить в предположении о наличии ошибки в одной работе проекта.

2.3 Критерий оптимальности и постановка задачи выбора контрольных точек в особо ответственных проектах

Для сравнения наборов контрольных точек с целью определения оптимального или, по крайней мере, рационального для данного проекта набора необходимо определить критерии сравнения. Эти критерии должны опираться на целевые функции, адекватно отражающие зависимости потерь при ошибках в работах от времени, которое потребуется на их устранение, при различных наборах контрольных точек. Как указано в параграфе 1.3, возможными являются как линейная, так и нелинейная зависимость потерь от времени восстановления правильности выполнения проекта. При этом для особо ответственных проектов нелинейная зависимость является более характерной. Критерий оптимальности для ООП предложен автором в работе [Марон А. И., Марон М. А., 2012].

Рассмотрим ООП, диагностической модели которого соответствует определённый граф. Каждому набору контрольных точек S соответствует своё разбиение множества всех работ проекта на подмножества работ, ошибки в которых неразличимы. Основываясь на ДМП, можно сказать по-другому, что каждому набору контрольных точек S соответствует своё

разбиение графа проекта на подграфы $\{G\}_s$, состоящие из вершин, соответствующих работам, с точностью до которых этот набор локализует возможные ошибки. Так, для проекта, ДМП которого представлена на рисунке 2.1, набору контрольных точек $S = \{П^4; П^6\}$ соответствует набор подграфов $\{G\}_s$ = {(1,2,3,4); (5,6); (7,8)}.

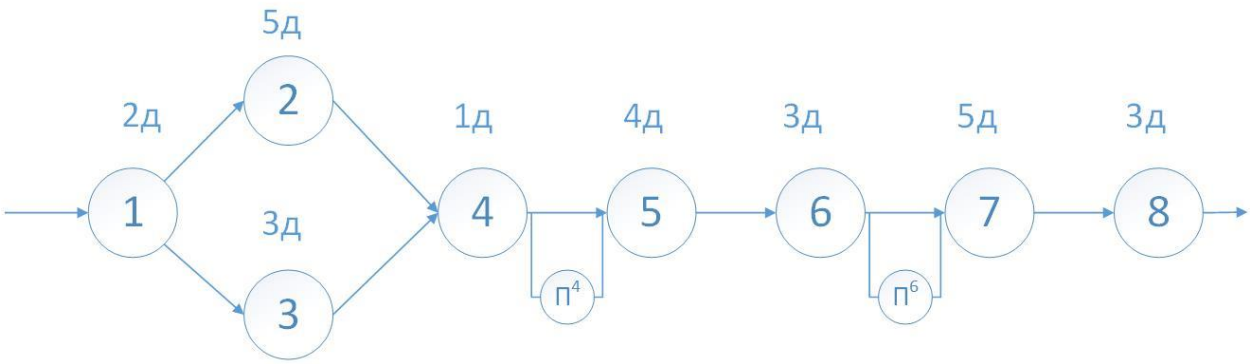

Рис. 2.1. Пример ООП с двумя контрольными точками

При данном наборе контрольных точек ошибки в работах, которым соответствуют вершины одного подграфа $G \epsilon \{G\}_s$, не различимы. Если будет допущена ошибка при выполнении любой работы $i \epsilon G \epsilon \{G\}_s$, то, в соответствии с основным свойством ПССР, для восстановления правильности выполнения проекта придётся переделать все работы, которым соответствуют вершины подграфа G. В дальнейшем для краткости будем говорить, что именно работы образуют подграф. Как было отмечено в параграфе 1.3, время t_i', затрачиваемое на переделку работы в ПССР, может отличаться от планового времени её выполнения, как правило, в большую сторону; а может и равняться ему. Для того, чтобы не усложнять обозначения, ограничимся рассмотрением случая, когда $t_i' = t_i$.

Как указано в параграфе 1.3, для ООП характерны нелинейные зависимости потерь, вызванных задержкой окончания ООП, от времени восстановления правильности выполнения проекта. С точностью до

коэффициентов для ООП характерной является следующая зависимость потерь от времени восстановления правильности выполнения проекта:

$$L(T) = \begin{cases} 0 & \text{при } T < t_д \\ b * c^{\beta*(T-t_д)} & \text{при } T \geq t_д \end{cases}, \quad (2.8)$$

где T _– время восстановления правильности выполнения проекта;

$L(\mathrm{T})$ – потери от задержки окончания проекта на время T;

$t_д$ – время допустимой задержки;

b, c, β – коэффициенты, причём $c > 1$; b и β положительные.

Заметим, что именно так, в частности, зависит суммарная задержка поездов на участке от времени проведения работ по восстановлению или модернизации автоблокировки [Абрамов В., Разгонов А., Давлетьяров Б., 1977]. При этом коэффициенты зависят от интенсивности движения поездов.

Если потери нелинейно возрастают с увеличением времени восстановления правильности выполнения проекта, то среднее время восстановления уже не будет представительной целевой функцией. Представительной целевой функцией является среднее значение потерь, но для его вычисления недостаточно знать среднее время восстановления правильности выполнения проекта. Надо знать закон распределения этого времени, что даже для типовых проектов абсолютно нереально. Вместе с тем можно предложить целевую функцию для сравнения наборов контрольных точек, не использующую вероятностные оценки, но в то же время хорошо подходящую для характерной зависимости потерь от времени восстановления правильности выполнения ООП. Основой для этого является наличие времени допустимой задержки $t_д$. Если набор контрольных точек выбран так, что максимальное время восстановления правильности выполнения проекта не превосходит времени допустимой задержки, то потерь не будет. Конечно, это далеко не всегда возможно. Если время восстановления оказывается больше $t_д$, то с его ростом потери начинают

возрастать быстрее, чем при линейной зависимости. Максимальное время восстановления правильности выполнения проекта является тем параметром, который надо стремиться уменьшить, выбирая контрольные точки.

Так, для рассматриваемого примера максимальное время восстановления правильности выполнения проекта составит 11 дней при ошибке в одной из работ {1; 2; 3; 4}. При ошибке в одной из работ, входящих в этот подграф, значение потерь будет больше, чем при ошибках в работах, входящих в другие подграфы. Разбиение на подграфы определяется набором контрольных точек.

Поскольку потери от задержки окончания проекта быстро растут, то при диагностике особо ответственных проектов, в отличие от обычных (см. параграф 2.3), на рациональное количество контрольных точек не столь сильно влияют длительности проверок. Их количество обычно определяет руководитель проекта исходя из качественного анализа, основанного на технологических особенностях создаваемого продукта. Пусть количество контрольных точек m задано. Тогда обоснованным является следующее определение оптимального набора контрольных точек.

Определение. Будем считать, что при нелинейной зависимости потерь от времени восстановления оптимальным является набор из m контрольных точек S^*, который разбивает множество всех работ проекта на подмножества таким образом, что соответствующее ему максимальное время восстановления правильности выполнения проекта минимально – меньше, чем при любом другом выборе m контрольных точек в данном проекте:

$$S^* \rightarrow \min_S \max_{G \in \{G\}_S} T_G = \min_S \max_{G \in \{G\}_S} \sum_{i \in G} t_i \quad (2.9),$$

где S^{*-} оптимальный набор контрольных точек;

S – множество возможных наборов контрольных точек;

G – подграф, состоящий из вершин, соответствующих работам, с точностью до которых определяется ошибка;

$\{G\}_S$ – множество подграфов G, соответствующее набору S;

T_G – время восстановления правильности выполнения проекта при определении ошибки с точностью до G.

i – номер вершины подграфа G, соответствующий работе проекта с данным номером;

t_i - время исправления ошибки в работе i, которое, напомним, считаем равным плановой длительности этой работы.

Изложенное позволяет сделать следующие выводы.

1. Впервые предложен критерий для сравнения наборов контрольных точек в ООП для случая, когда потери от задержки проекта нелинейно зависят от времени восстановления правильности выполнения проекта.

2.4 Методы выбора контрольных точек в особо ответственных проектах

Задача выбора контрольных точек в ООП допускает следующую формальную постановку [Марон А. И., Марон М. А., 2012].

Постановка задачи

Имеется ООП с работами, сильно связанными по результатам (ПССР). При выполнении работ проекта осуществляется поэлементный контроль. Задан сетевой график проекта. Соответствующий ему граф состоит из $n+1$ вершины, в котором одна вершина (с номером $n+1$) конечная, соответствует последней работе проекта. Для проверки правильности выполнения работ можно установить $m \ll n$ контрольных точек для выполнения промежуточных проверок. Окончательная проверка после выполнения последней работы проекта выполняется обязательно. Длительность работы с номером i равна $t_i (i = 1,2, \dots, n+1)$.

Требуется определить набор контрольных точек, ориентируясь на минимаксный критерий оптимальности.

Поставленная задача выбора контрольных точек в особо ответственных проектах может быть решена полным перебором, как и любая другая комбинаторная задача оптимизации. Однако проблема размерности не позволяет осуществить такой перебор для реальных проектов. Применению метода динамического программирования и метода ветвей и границ препятствует неаддитивное изменение целевой функции, стоящей в правой части (2.9), при добавления очередной контрольной точки к ранее выбранным.

Предлагается эвристический подход к решению поставленной задачи, основанный на идеях теории информации К. Шеннона.

Будем считать, что если при окончательной проверке получен отрицательный результат, то это является следствием неправильного выполнения одной и только одной работы проекта. Данное предположение сделано исходя из того, что при выполнении работ проекта осуществляется поэлементный контроль. Будем считать, что длительность работы отражает её сложность и, будем использовать формулу (2.7) для определения вероятности p_i наличия ошибки в работе i.

Создадим диагностическую модель (ДМП) рассматриваемого проекта. Тогда проверка $П^j$, осуществлённая после выполнения работы j $(j = 1, 2, \dots, n)$, имеет положительный результат π_1^j тогда и только тогда, когда работа j выполнена правильно и правильно выполнены все работы, ей предшествующие. В противном случае проверка $П^j$ будет иметь отрицательный результат π_0^j, означающий, что либо работа j выполнена неправильно, либо неправильно выполнена одна и только одна работа, ей предшествующая. Это следует из того, что по условию проект состоит из

работ, сильно связанных по результатам. Зная вероятности ошибок и на основании ДМП, можно предложить эвристические методы выбора контрольных точек. В основе их лежит теория информации К. Шеннона. Использование теории информации в данном случае основано на том, что проверки – это возможные эксперименты. Надо выбирать те эксперименты из числа возможных, которые дают наибольшее количество информации об объекте исследования.

2.4.1 Метод последовательного условного выбора контрольных точек в особо ответственных проектах

Принципиально невозможно применить метод динамического программирования или метода ветвей и границ для решения поставленной задачи. Это связано с тем, что целевая функция (2.9) не является аддитивной. Более того, даже если принять в качестве целевой функции среднее время восстановления правильности выполнения проекта, то и она будет не аддитивна. Нельзя её значение, найденное для k точек, использовать для нахождения её значения при добавлении следующей точки.

Автором был предложен эвристический метод последовательного выбора контрольных точек [Марон М. А., 2012; Марон А. И., Марон М. А., 2010], основанный на информационном подходе К. Шеннона. Было предложено на каждом шаге выбирать контрольную точку так, чтобы она давала максимум информации о том, в какой из работ, правильность выполнения которых ещё не проверена, допущена ошибка. Поскольку каждая последующая проверка выбирается при условии, что ранее выбранные проверки дали положительные результаты, метод назван «методом последовательного условного выбора контрольных точек». Этот метод можно назвать и односторонним, поскольку проверенные работы исключаются из перечня возможных мест проведения проверок. Каждая последующая проверка выбирается как-бы «вперёд» по ходу реализации

проекта. «Движение вперёд» отражает особенности проекта как объекта диагноза и отличает этот метод от известных в технической диагностике.

Этот метод состоит в следующем.

Введём случайную величину N – номер работы, выполненной неправильно. По условию задачи она может принять одно из значений $\{1;\ 2;\ \ldots;\ n;\ (n+1)\}$. Тогда случайной величине N можно поставить в соответствие энтропию:

$$H(N) = -\sum_{i=1}^{n+1} p_i \, log_2 \, p_i, \qquad (2.10)$$

где p_i – вероятность ошибки в работе i, вычисляемая через плановую длительность t_i по формуле (2.7).

Введём случайную величину π^j результат проверки с номером j. Как сказано в условии, она может принять одно из двух значений π_0^{j} или π_1^{j}. Причём вероятности этих значений зависят от того, какие работы предшествуют работе j. Осуществление проверки Π^j уменьшает неопределённость относительно того, какая работа выполнена неправильно. Соответствующую информацию можно найти как меру уменьшения неопределённости по формуле:

$$I\left(N;\pi^j\right) = H(N) - H(N/\pi^j), \qquad (2.11)$$

где $I\left(N;\pi^j\right)$ – средняя взаимная информации случайных величин N и π^j;

$H(N)$ – энтропия случайной величины N, вычисляемая по формуле (2.10);

$H(N/\pi^j)$ – условная (остаточная) энтропия.

Условная энтропия $\mathrm{H(N/\pi^j)}$ является средней мерой неопределённости, которая останется при условии, что проверка будет осуществляться после выполнения работы j.

Можно доказать, что количество информации $\mathrm{I(N;\pi^j)}$ достигает максимума, если выполняется проверка, неопределённость результата

которой – $H(\pi^j)$ – максимальна. Ей соответствует минимальная остаточная энтропия.

Значение $H(\pi^j)$ можно найти по формуле:

$$H(\pi^j) = -P(\pi_0^j) log_2 P(\pi_0^j) - P(\pi_1^j) log_2 P(\pi_1^j). \qquad (2.12)$$

Здесь

$P(\pi_0^j)$ – вероятность отрицательного результата π_0^j проверки $П^j$, она равна сумме вероятностей неправильного выполнения работы j и всех работ, ей предшествующих:

$$P(\pi_0^j) = \sum_{i \in G(\pi_0^j)} p_i, \qquad (2.13)$$

где $G(\pi_0^j)$ – подмножество вершин графа проекта G, соответствующих работам, при ошибках в которых проверка $П^j$ будет иметь отрицательный результат π_0^j;

$P(\pi_1^j)$ – вероятность положительного результата π_1^j проверки $П^j$, она дополняет $P(\pi_0^j)$ до единицы.

Нахождение первой промежуточной проверки $П^{r_1}$, имеющей максимальную энтропию, осуществляем при условии, что проверка после последней работы проекта $П^{(n+1)}$ обязательно проводится и имеет отрицательный результат.

Правило её нахождения следующее:

$П^{r_1} \rightarrow$

$$max_{1 \le j \le n} H(\pi^j) = max_{1 \le j \le n} [-P(\pi_0^j) log_2 P(\pi_0^j) - P(\pi_1^j) log_2 P(\pi_1^j)]. \quad (2.13)$$

Для нахождения $П^{r_1}$ надо провести n расчётов по формуле (2.12) и найти максимум. При этом процедура расчёта энтропии проверки по формуле (2.12) проще, чем процедура расчёта информации по формуле (2.11), что позволяет существенно сократить время расчёта.

Рассмотрим ситуацию, которая возникнет, когда после реализации работы r_1 будет выполнена проверка $П^{r_1}$. Если будет получен отрицательный результат, то придётся осуществить поиск неправильно выполненной работы среди подмножества работ, неправильное выполнение одной из которых приводит к этому результату. Обозначим множество номеров таких работ через $G(\pi_0^{r_1})$, поскольку, согласно постановке, проект относится к классу ПССР. Все эти работы подлежат переделыванию. После чего можно считать, что в работах, входящих в $G(\pi_0^{r_1})$, ошибок нет.

Если будет получен положительный результат проверки, то это означает, что работа r_1 и все работы, ей предшествующие, то есть все работы с номерами из G_0^1, выполнены правильно. Обозначим через $G(\pi_1^{r_1})$ множество номеров работ, не входящих в $G(\pi_0^{r_1})$. Именно среди этих работ необходимо искать ту, после которой надо выполнить следующую проверку. Для того чтобы вычислить энтропию результата проверки работы, входящей в $G(\pi_1^{r_1})$, при условии, что проверка работы с номером r_1 выполнена, необходимо для каждой работы i, входящей в $G(\pi_1^{r_1})$, определить условную вероятность неправильного выполнения с учётом того, что неправильно могут быть выполнены только работы с номерами из $G(\pi_1^{r_1})$. Эти условные вероятности $P(i/\pi_1^{r_1})$ можно найти по формуле:

$$P(i/\pi_1^{r_1}) = \frac{p_i}{P\left(G(\pi_1^{r_1})\right)} = \frac{t_i}{T\left(G(\pi_1^{r_1})\right)}, \qquad (2.14)$$

где $P\left(G(\pi_1^{r_1})\right)$ – сумма исходных вероятностей неправильного выполнения работ, входящих в $G(\pi_1^{r_1})$;

$T\left(G(\pi_1^{r_1})\right)$ – сумма длительностей работ, входящих в $G(\pi_1^{r_1})$.

Формула (2.14) следует из формулы Байеса с учётом формулы (2.7).

На основании вышеизложенного можно предложить следующий метод последовательного выбора работ проекта, после которых надо осуществлять проверки:

$$\Pi^{r_1} \rightarrow$$

$$max_{1\le j\le n} H(\pi^j) = max_{1\le j\le n}[-P(\pi_0^j)log_2P(\pi_0^j) - P(\pi_1^j)log_2P(\pi_1^j)],$$

$$\Pi^{r_2} \rightarrow \max_{j\in G(\pi_1^{r_1})} H(\pi^j/\pi_1^{r_1}),$$

$$\Pi^{r_3} \rightarrow \max_{j\in G(\pi_1^{r_1};\pi_1^{r_2})} H(\pi^j/\pi_1^{r_1};\pi_1^{r_2}),$$

$$\ldots,$$

$$\Pi^{r_k} \rightarrow \max_{j\in G(\pi_1^{r_1};\pi_1^{r_2};\ldots;\pi_1^{r_{k-1}})} H(\pi^j/\pi_1^{r_1};\pi_1^{r_2};\ldots;\pi_1^{r_{k-1}}), \quad (2.15)$$

$$\ldots,$$

$$\Pi^{r_m} \rightarrow \max_{j\in G(\pi_1^{r_1};\pi_1^{r_2};\ldots;\pi_1^{r_k};\ldots;\pi_1^{r_m})} H(\pi^j/\pi_1^{r_1};\pi_1^{r_2};\ldots;\pi_1^{r_k};\ldots;\pi_1^{r_m}).$$

В (2.15):

$H(\pi^j/\pi_1^{r_1};\pi_1^{r_2};\ldots;\pi_1^{r_{k-1}})$ – энтропия результата проверки π^j при условии, что конечная проверка Π^{n+1} показывает наличие ошибки в проекте, а проверки $\pi_1^{r_1};\ \pi_1^{r_2};\ldots;\ \pi_1^{r_{k-1}}$ имеют положительные результаты;

$G(\pi_1^{r_1};\ \pi_1^{r_2};\ldots;\ \pi_1^{r_{k-1}})$ – множество работ, при ошибках в которых указанные выше проверки будут иметь положительные результаты.

Энтропию $H(\pi^j/\pi_1^{r_1};\pi_1^{r_2};\ldots;\pi_1^{r_{k-1}})$ можно найти по формуле:

$$H(\pi^j/\pi_1^{r_1};\pi_1^{r_2};\ldots;\pi_1^{r_{k-1}}) =$$

$$= -P\left(\frac{\pi_0^j}{\pi_1^{r_1};\pi_1^{r_2}};\ldots;\pi_1^{r_{k-1}}\right)log_2P\left(\frac{\pi_0^j}{\pi_1^{r_1};\pi_1^{r_2}};\ldots;\pi_1^{r_{k-1}}\right)$$

$$-\left[1-P\left(\frac{\pi_0^j}{\pi_1^{r_1};\pi_1^{r_2}};\ldots;\pi_1^{r_{k-1}}\right)\right]log_2\left[1-P\left(\frac{\pi_0^j}{\pi_1^{r_1};\pi_1^{r_2}};\ldots;\pi_1^{r_{k-1}}\right)\right].$$

Здесь

$$P\left(\pi_0^j/\pi_1^{r_1};\pi_1^{r_2};\dots;\pi_1^{r_{k-1}}\right)=\sum_{i\in G\left(\pi_0^j/\pi_1^{r_1};\pi_1^{r_2};\dots;\pi_1^{r_{k-1}}\right)} P(i/\pi_1^{r_1};\pi_1^{r_2};\dots;\pi_1^{r_{k-1}}),$$

и

$$P(i/\pi_1^{r_1};\pi_1^{r_2};\dots;\pi_1^{r_{k-1}})=\frac{p_i}{P\left(G\left(\pi_1^{r_1};\pi_1^{r_2};\dots;\pi_1^{r_{k-1}}\right)\right)}=\frac{t_i}{T\left(G\left(\pi_1^{r_1};\pi_1^{r_2};\dots;\pi_1^{r_{k-1}}\right)\right)},$$

$$P\left(G\left(\pi_1^{r_1};\pi_1^{r_2};\dots;\pi_1^{r_{k-1}}\right)\right)=\sum_{i\in G\left(\pi_1^{r_1};\pi_1^{r_2};\dots;\pi_1^{r_{k-1}}\right)} p_i\ ,$$

$$T\left(G\left(\pi_1^{r_1};\pi_1^{r_2};\dots;\pi_1^{r_{k-1}}\right)\right)=\sum_{i\in G\left(\pi_1^{r_1};\pi_1^{r_2};\dots;\pi_1^{r_{k-1}}\right)} t_i\ ,$$

где $G\left(\frac{\pi_0^{\mathrm{j}}}{\pi_1^{\mathrm{r}_1};\pi_1^{\mathrm{r}_2}};\dots;\pi_1^{\mathrm{r}_{k-1}}\right)$ – множество работ, при ошибках в которых проверка $П^j$ будет иметь отрицательный результат при условии, что все ранее выполненные проверки дали положительные результаты $\pi_1^{\mathrm{r}_1};\pi_1^{\mathrm{r}_2};\dots;\pi_1^{\mathrm{r}_{k-1}}$; или, что то же – $G\left(\frac{\pi_0^{\mathrm{j}}}{\pi_1^{\mathrm{r}_1};\pi_1^{\mathrm{r}_2}};\dots;\pi_1^{\mathrm{r}_{k-1}}\right)$, это подмножество работ из $G(\pi_1^{\mathrm{r}_1};\pi_1^{\mathrm{r}_2};\dots;\pi_1^{\mathrm{r}_{k-1}})$, при которых проверка $П^j$ будет иметь отрицательный результат;

$\mathrm{G}(\pi_1^{\mathrm{r}_1};\pi_1^{\mathrm{r}_2};\dots;\pi_1^{\mathrm{r}_{k-1}})$ – множество работ, при ошибках в которых указанные выше проверки будут иметь положительные результаты.

Соответственно предлагается следующий метод последовательного условного выбора контрольных точек в особо ответственных проектах.

1. Для всех $j=1,2,\dots,n$ вычислим энтропию результата проверки $П^j$ по формуле (2.12). Определим в соответствии с (2.13) номер проверки r_1,

энтропия результата которой максимальна. Примем эту проверку в качестве первой контрольной точки в проекте.

2. Удалим из дальнейшего рассмотрения подмножество работ проекта, состоящее из работы r_1 и всех работ, ей предшествующих. Останется подмножество работ с номерами, принадлежащими $G(\pi_1^{r_1})$. Осуществим перерасчёт вероятностей по формуле (2.14).

3. Для всех проверок с номерами, принадлежащими $G(\pi_1^{r_1})$, выполним расчёт энтропий результатов. Определим номер проверки, энтропия результата которой максимальна. Примем её в качестве второй контрольной точки.

4. Следуя решающему правилу (2.15), выполним действия, аналогичные указанным в пунктах 3 и 4, осуществляя расчёт энтропии проверок по формуле (2.16), пока не будут выбраны все m контрольных точек.

Таким образом, впервые предложен метод рационального выбора контрольных точек в ООП. Он основан на идеях теории информации К. Шеннона и предложенных автором диагностической модели проекта и методе расчёта вероятностей ошибок, возникающих в работах ООП. Разработана программная реализация метода для проектов, управление которыми осуществляется с применением MS – Project. Подробнее об этом в третьей главе диссертации.

2.4.2 Модифицированный метод последовательного условного выбора контрольных точек в особо ответственных проектах

Метод, изложенный в параграфе 2.4.1, основан исключительно на информационном подходе к выбору проверок. Соответственно набор проверок будет предопределён топологией графа работ проекта и распределением вероятностей. В результате первая проверка осуществит половинное разбиение графа работ по вероятностям. Следующая будет

осуществлять подобное разбиение для подграфа работ, не являющихся логическими предшественниками работы, которой соответствует первая проверка и. т. д. В результате до первой проверки может оказаться уже выполненным слишком большой объём работ проекта. Если после проверки окажется, что какая-то работа выполнена неправильно, то суммарное время поиска и устранения ошибки может оказаться большим. Исходя из этого, в работе был предложен и модифицированный метод последовательного условного выбора контрольных точек [Марон А. И., Марон М. А., 2012; Марон М. А., 2014]. Он состоит в следующем.

Будем последовательно выбирать место проведения каждой проверки, исходя из значения критерия оценки эффективности. Критерий оценки эффективности очередной проверки (контрольной точки) построим, основываясь на энтропии её результата и суммарном времени работ, которым соответствуют вершины подграфа, с точностью до которого при отрицательном результате этой проверки будет локализована работа, содержащая ошибку, при условии, что все предыдущие проверки дали положительные результаты.

Рассмотрим очередной шаг k выбора контрольной точки. Допустим, что на предыдущих шагах 1, 2, …, k-1 уже определён набор контрольных точек $\pi_1^{r_1}; \pi_1^{r_2}; \ldots; \pi_1^{r_{k-1}}$. Рассмотрим проверку Π^j , которую можно выполнить на этапе k. Энтропию её результата $H\left(\pi^j/\pi_1^{r_1}; \pi_1^{r_2}; \ldots; \pi_1^{r_{k-1}}\right)$ можно вычислить по формуле (2.16). Энтропия – положительная характеристика проверки: чем она больше, тем больше информации даёт проверка. С другой стороны, при отрицательном результате данной проверки ошибка будет локализована с точностью до множества работ $G(\pi_0^j/\pi_1^{r_1}; \pi_1^{r_2}; \ldots; \pi_1^{r_{k-1}})$. При ошибке в любой из работ из этого множества для восстановления правильности выполнения проекта все работы, входящие в него, придётся переделать. На это понадобится время:

$$T\left(\mathrm{G}\left(\pi_0^{\mathrm{j}}/\pi_1^{\mathrm{r}_1};\pi_1^{\mathrm{r}_2};\ldots;\pi_1^{\mathrm{r}_{k-1}}\right)\right)=\sum_{i\in G\left(\pi_0^{\mathrm{j}}/\pi_1^{\mathrm{r}_1};\pi_1^{\mathrm{r}_2};\ldots;\pi_1^{\mathrm{r}_{k-1}}\right)}t_i\,, \qquad (2.17)$$

где

t_i – длительность работы i.

Время (2.17) – негативная характеристика проверки: чем это время больше, тем большее время потребуется для восстановления правильности выполнения проекта при локализации ошибки.

Примем в качестве критерия оценки эффективности очередной возможной проверки Π^j дробь W_k^j **,** числителем которой является энтропия результата проверки Π^j, а в знаменателе стоит функция от $T(\pi_0^j/\pi_1^{r_1};\pi_1^{r_2};\ldots;\pi_1^{r_{k-1}})$. Определим вид этой функции. Требования к ней таковы:

1) это возрастающая положительная функция;

2) скорость её изменения существенно меньше скорости изменения времени $T(\pi_0^{\mathrm{j}}/\pi_1^{\mathrm{r}_1};\pi_1^{\mathrm{r}_2};\ldots;\pi_1^{\mathrm{r}_{k-1}})$;

3) в нуле значение функции равно единице.

Первое из этих требований следует из того, что значение должно уменьшаться при росте $T(\pi_0^j/\pi_1^{r_1};\pi_1^{r_2};\ldots;\pi_1^{r_{k-1}})$. Второе требование связано с необходимостью по возможности уравнять диапазон изменения числителя и знаменателя, а также обеспечить устойчивость критерия выбора к малым ошибкам в определении времени работ. Третье требование обеспечит эквивалентность этого подхода и правила, предложенного в 2.4.1 для случая, когда время не учитывается.

Всем трём условиям удовлетворяет функция:

$$F\left(T(\pi_0^j/\pi_1^{r_1};\pi_1^{r_2};\ldots;\pi_1^{r_{k-1}})\right)=1+log_2(1+T(\pi_0^j/\pi_1^{r_1};\pi_1^{r_2};\ldots;\pi_1^{r_{k-1}})) \qquad (2.18)$$

Соответственно, предлагается следующий критерий оценки эффективности возможной проверки Π^j на шаге k:

$$W_k^j = \frac{H\left(\pi^j/\pi_1^{r_1};\pi_1^{r_2};\ldots;\pi_1^{r_{k-1}}\right)}{1+log_2(1+T(\pi_0^j/\pi_1^{r_1};\pi_1^{r_2};\ldots;\pi_1^{r_{k-1}}))} \quad (2.19)$$

Нахождение первой промежуточной проверки Π^{r_1} осуществляем при условии, что проверка после последней работы проекта Π^{n+1} обязательно проводится и имеет отрицательный результат. Правило её нахождения следующее:

$$\Pi^{r_1} \to max_{1\le j\le n} W_1^j = \max_{1\le j\le n} \frac{H(\pi^j)}{1+log_2(1+T\left(\pi_0^j\right))} \quad (2.20)$$

где $H\left(\pi^j\right)$ – энтропия результата проверкиΠ^j ;

$T\left(\pi_0^j\right)$ – сумма длительностей всех работ, при которых проверка Π^j имеет отрицательный результат π_0^j

Проверку Π^{r_k}, которую следует включить в набор контрольных точек на шаге k $(k = 2,3, \ldots m)$, будем находить по решающему правилу:

$$\Pi^{r_k} \to max_{j\in G(\pi_1^{r_1};\pi_1^{r_2};\ldots;\pi_1^{r_{k-1}})} W_k^j. \quad (2.21)$$

В (2.21) значение W_k^j рассчитывается по формуле (2.19), при этом числитель определяется по формуле (2.16), а значение времени, стоящее в знаменателе, по формуле (2.17).

Предлагается следующий модифицированный метод последовательного условного выбора контрольных точек в особо ответственных проектах. Он состоит в следующем.

1. Первую контрольную точку определим на основании решающего правила (2.20).

2. Последующие контрольные точки определим на основании решающего правила (2.21) для $k = 2, 3, \dots, m$.

Автором разработана программная реализация предложенного метода для проектов, управление которыми осуществляется с применением MS – Project. Подробнее об этом в третьей главе диссертации.

2.4.3 Метод последовательного безусловного выбора контрольных точек в ООП

В изложенных выше методах последовательного условного выбора контрольных точек в ООП каждая следующая контрольная точка выбиралась так, чтобы она давала максимум информации при условии, что все проверки в предшествующих точках имеют положительные результаты. Автором предложен и безусловный метод последовательного выбора контрольных точек. В соответствии с ним на каждом шаге выбирается та контрольная точка, при которой набор, состоящий из неё и ранее выбранных контрольных точек, даёт максимум информации о том, в какой работе проекта допущена ошибка. Этот метод сложнее с вычислительной точки зрения, чем ранее предложенные условные методы. Перейдём к его изложению.

Будем выбирать контрольные точки последовательно – шаг за шагом. Допустим, что после шага $k-1$ $(k = 2, 3, .., m)$ получен набор контрольных точек $S_{к-1} = \{\Pi^{r_1}; \Pi^{r_2}; \dots, \Pi^{r_{k-1}}\}$. Введём случайную величину $s_{k-1} = \{\pi_1^{r_1}; \pi_1^{r_2}; \dots; \pi_1^{r_{k-1}}\}$ – показание контрольных точек набора $S_{к-1}$. На

следующем шаге k добавим в набор ту контрольную точку Π^{r_k}, при которой будет получен такой набор контрольных точек:

$$S_k = S_{k-1} \cup \Pi^{r_k},$$

что

$$\Pi^{r_k} \to I(N, s_k) = \max_{j \notin S_{k-1}} I(N, s_{к-1} \cup \pi^j), \quad (2.22)$$

где $I\left(N, s_{k-1} \cup \pi^j\right)$ количество информации о номере N работы проекта, выполненной с ошибкой, которое даёт совокупность результатов $\{\pi_1^{r_1}; \pi_1^{r_2}; \dots; \pi_1^{r_{k-1}}; \pi^j\}$ проверок в контрольных точках $S_{k-1} \cup \Pi^j = \{\Pi^{r_1}; \Pi^{r_2}; \dots, \Pi^{r_{k-1}}; \Pi^j\}$.

По определению информация $I(N, s_k)$ равна:

$$I(N, s_k) = H(N) - H(N/s_k), \quad (2.23)$$

где $H(N/s_k)$ – условная энтропия (среднее значение остаточной неопределённости) случайной величины N при возможных результатах $s_к$ выбранных проверок.

Непосредственное вычисление условной энтропии, основанное на её определении по Шеннону, связано с необходимостью генерации всех допустимых значений величины s_k и расчёте соответствующих сумм произведений условных вероятностей на их двоичные логарифмы. Это может вызвать огромные вычислительные трудности. Вместе с тем в данном случае вычисление условной энтропии можно упростить, основываясь на том, что неопределённость создают только те работы проекта, ошибки в которых неразличимы при заданном наборе контрольных точек.

Набор контрольных точек S_k разбивает множество всех работ G_0 проекта на два подмножества. Одно из них $G_u(s_k)$ состоит из работ, ошибки в которых по совокупности результатов s_k этих проверок определяются однозначно – с точностью до работы, выполненной с ошибкой. Каждой

ошибке в работе $i \in G_u(s_k)$ соответствует своё уникальное сочетание результатов проверок. Второе подмножество $G_{\text{а}}(s_k)$ состоит из подмножеств работ $\{G_f\}$, ошибки в которых не различимы. Всем работам $i \in G_f$ соответствует одно и то же сочетание результатов проверок. При этом ошибки в работах, входящие в разные подмножества, различимы.

Остаточную энтропию, входящую в формулу (2.23), можно вычислить по формуле:

$$H(N/s_k) = -\sum_{i \in G_u(s_k)} p_i \, log_2 \, p_i + \sum_{G_f \in G_u(s_k)} \sum_{i \in G_f} p_i \, log_2 \sum_{i \in G_f} p_i. \quad (2.24)$$

Подставив значение энтропий (2.10) и (2.24) в (2.23), получим:

$$I(N, s_k) = -\sum_{i \in G_u(s_k)} p_i \, log_2 \, p_i - \sum_{G_f \in G_a(s_k)} \sum_{i \in G_f} p_i \, log_2 \sum_{i \in G_f} p_i, \quad (2.25)$$

где $I(N, s_k)$ – количество информации о номере N работы проекта, выполненной с ошибкой, которое даёт совокупность результатов s_k;

p_i – вероятность ошибки в работе i;

$G_u(s_k)$ – подмножество работ проекта, ошибки в которых определяются однозначно;

$G_{\text{а}}(s_k)$ – подмножество работ проекта, ошибки в которых определяются не однозначно,

G_f – подмножество работ, ошибки в которых неразличимы.

Формула (2.25) позволяет ясно выразить, что даёт установка контрольных точек в проекте в информационном аспекте. Количество информации, которое дают контрольные точки, складывается из двух слагаемых. Первое из них показывает, насколько уменьшилась неопределённость в результате возможности однозначного определения ошибок в ряде работ. Второе показывает, насколько уменьшилась

неопределённость за счёт того, что работы, ошибки в которых не могут быть определены однозначно, локализованы с точностью до меньшего числа работ, чем если бы контрольных точек в проекте не было.

Метод выбора контрольных точек заключается в следующем.

1. Приняв $S_0 \equiv \emptyset$, выбрать первую контрольную точку так, чтобы она давала максимальное количество информации (2.22).

2. На каждом последующем шаге k, начиная со второго и до шага m, добавлять в набор ту контрольную точку, при которой информация (2.22) достигает максимума.

Автором разработана программная реализация предложенного метода для проектов, управление которыми осуществляется с применением MS – Project. Для оценки его эффективности проведена представительная серия расчётных экспериментов. Подробнее об этом в третьей главе диссертации.

2.4.4 Выбор контрольных точек в мультисценарных ООП

Основной чертой проекта является уникальность состава работ. Вместе с тем план проекта создаётся заранее. Естественно желание предусмотреть возможные изменения состава работ. Стандартная математическая модель проекта не предусматривает сценариев, но они необходимы для того, чтобы учесть при планировании проекта возможное изменения состава работ на стадии его реализации [Williams T. and others. 1995]. Этим вопросам посвящён GERT-анализ [Pritsker A. A. B., Happ W. W., 1966; Pritsker A. A. B., Whitehouse G. E., 1966; Pritsker A. A. B., 1974]. Однако решённые им вопросы не затрагивали до сих пор проблему выбора точек контроля. Вместе с тем на мультисценарные ООП можно обобщить метод, изложенный в разделе 2.4.3. Впервые это сделано автором в работе [Марон М. А., 2016].

Постановка задачи выбора контрольных точек в мультисценарном ООП несколько отличается от приведенной выше постановки задачи выбора

контрольных точек в ООП, при реализации которого сценарии не предполагаются. Это отличие связано с тем, что каждому сценарию соответствует свой граф проекта и эти графы имеют общие вершины.

Постановка задачи выбора контрольных точек в мультисценарном ООП

Имеется особо ответственный проект (ООП) с работами, сильно связанными по результатам (ПССР). При выполнении работ проекта осуществляется поэлементный контроль. Известно, что в конкретной реализации он может проходить по различным сценариям. Каждому сценарию y можно поставить в соответствие свой план, отображаемый графом G_y ($y = 1,2,\dots,L$). В дальнейшем будем называть их сценарными графами. Сценарий y может реализоваться с вероятностью P_y. Сценарные графы имеют общие вершины. Для простоты будем считать, что имеется одна общая выходная вершина. Работа, ей соответствующая, выполняется при любом сценарии и является заключительной запланированной работой проекта. Работы и соответствующие им вершины пронумерованы так, что каждая работа получает номер один раз, даже если она входит в несколько сценарных графов. Всего имеется $n + 1$ работ (вершин). Для проверки правильности выполнения работ можно установить $m \ll n$ контрольных точек для выполнения промежуточных проверок. Окончательная проверка после выполнения последней работы проекта выполняется обязательно. Длительность работы с номером i равна t_i ($i = 1,2,\dots,n+1$).

Требуется определить набор контрольных точек, ориентируясь на критерий оптимальности (2.9).

На рисунке 2.2 приведен пример проекта, при реализации которого возможны два сценария, которым соответствуют графы $G_1 = \{1; 2; 3; 4; 5\}$ и $G_2 = \{1; 2; 6; 7; 4; 5\}$.

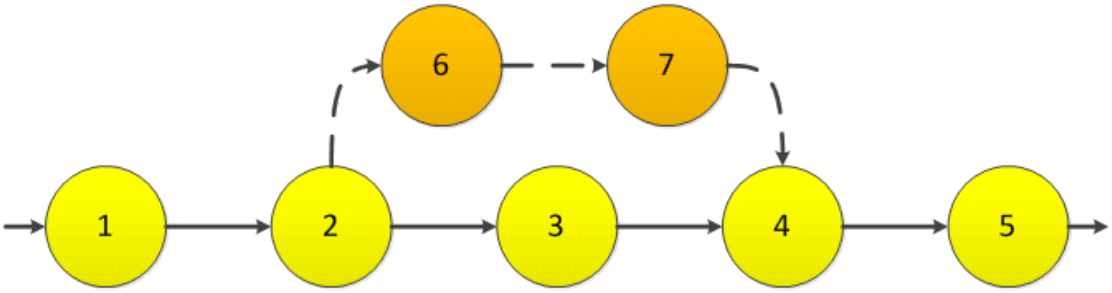

Рис.2.2. Условный проект с двумя возможными сценариями реализации

Введём случайные величины N – номер работы, выполненной с ошибкой, Y – номер сценария, по которому будет реализован проект.

Будем определять контрольные точки последовательно. Допустим, что к шагу k уже определён набор контрольных точек $S_{к-1} = \Pi^{r_1}; \Pi^{r_2}; \dots, \Pi^{r_{k-1}}\{.$ Добавим к нему ту контрольную точкуΠ^{r_k}, при которой полученному набору $S_к$ соответствует минимум остаточной энтропии:

$$\Pi^{r_k} \to min_{j \notin S_{k-1}} H[(N/Y), s_{k-1} \cup \pi^j] = min_{j \notin S_{k-1}} \sum_{y=1}^{L} P_y H[(N/y), s_{k-1} \cup \pi^j], \quad (2.26)$$

где $H[(N/y), s_{k-1} \cup \pi^j]$ – остаточная энтропия номера работы, выполненной неправильно при сценарии y, с учётом возможных показаний $\{\pi_1^{r_1}; \pi_1^{r_2}; \dots; \pi_1^{r_{k-1}}; \pi^j\}$ набора проверок $\{\Pi^{r_1}; \Pi^{r_2}; \dots, \Pi^{r_{k-1}}; \Pi^j\}$.

Для вычисления по формуле (2.23) потребуются знать p_{iy} – вероятность ошибки в работе i при реализации проекта по сценарию y. По аналогии с (2.7) будем считать, что:

$$p_{iy} = \frac{t_i}{\tau_y}, \quad (2.27)$$

где τ_y – суммарное время выполнения работ, входящих в G_y.

Ненамного сложнее, чем по правилу (2.23), можно определять очередную контрольную точку, ориентируясь непосредственно на среднюю информацию:

$$\Pi^{r_k} \to max_{j \notin S_{k-1}} I[(N/Y), s_{k-1} \cup \pi^j] = max_{j \notin S_{k-1}} \sum_{y=1}^{L} P_y \, I[(N/y), s_{k-1} \cup \pi^j], \quad (2.28)$$

где $I[(N/y), s_{k-1} \cup \pi^j]$ – среднее количество информации, содержащееся в возможных результатах $\{\pi_1^{r_1}; \pi_1^{r_2}; \ldots; \pi_1^{r_{k-1}}; \pi^j\}$ набора проверок $\{\Pi^{r_1}; \Pi^{r_2}; \ldots, \Pi^{r_{k-1}}; \Pi^j\}$ о номере работы, выполненной с ошибкой, при условии, что проект будет реализован по сценарию y.

Метод определения набора контрольных точек с помощью решающего правила (2.26) или решающего правила (2.28) аналогичен приведенному в 2.4.3.

При этом информацию $I[(N/Y), s_{k-1} \cup \pi^j] = I[(N/Y), s_k)$ можно вычислить по формуле, являющейся обобщением ранее выведенной формулы (2.25):

$$I[(N/Y), s_k)] = \sum_{y=1}^{L} P_y \left\{ - \sum_{i \in G_u(s_k, y)} p_{iy} \, log_2 \, p_{iy} - \sum_{G_f \in G_a(s_k, y)} \sum_{i \in G_f} p_{iy} \, log_2 \sum_{i \in G_f} p_{iy} \right\}, \quad (2.29)$$

где $I[(N/Y), s_k)]$ – количество информации о номере N работы проекта, выполненной с ошибкой, которое даёт совокупность результатов s_k при условии, что сценарий Y, по которому будет реализован проект, неизвестен;

P_y – вероятность сценария y;

p_{iy} – вероятность ошибки в работе i при реализации проекта по сценарию y, рассчитываемая по формуле (2.27);

y – номер возможного сценария, а L – количество возможных сценариев;

$G_u(s_k, y)$ – подмножество работ проекта, ошибки в которых однозначно определяются по результатам s_k при реализации проекта по сценарию y;

$G_{\text{a}y}(S_k)$ – подмножество работ проекта, ошибки в которых при реализации сценария y определяются неоднозначно;

G_f – подмножество работ, ошибки в которых неразличимы.

Таким образом, автором впервые предложены методы определения наборов контрольных точек в особо ответственных проектах. Теоретическим базисом этих методов является теория информации К. Шеннона. Проверки в контрольных точках отождествляются с экспериментами, целью которых является получение информации о работе, выполненной с ошибкой. Методы впервые изложены автором в работах [Марон А. И., Марон М. А., 2012] и [Марон М. А., 2014].

В главе 3 приведено описание программного комплекса, реализующего предложенные методы для проектов, управление которыми осуществляется с применением MS – Project. Методы не являются регулярными, то есть не гарантируют получения оптимального по критерию (2.9) набора контрольных точек, которому соответствует минимум максимального времени восстановления правильности выполнения проекта. Поэтому проведена представительная серия экспериментов, которая показала высокую эффективность комплексного использования предложенных методов. Подробнее об этом в главе 3.

Таким образом, в настоящем параграфе изложены следующие новые научные результаты диссертационного исследования.

1. Впервые предложены методы выбора контрольных точек в ООП. Методы используют предложенные автором диагностическую модель проекта (ДМП) и модель для расчёта вероятностей ошибок в ООП.

2. Методы не являются регулярными, но их научной базой является теория информации К. Шеннона, применение которой даёт хорошие результаты при планировании экспериментов, а проверки в контрольных точках являются именно экспериментами, выполняемыми в целях диагностирования проекта.

3. Метод последовательного условного выбора контрольных точек в ООП и модифицированный метод последовательного условного выбора контрольных точек в ООП существенно используют особенность проекта как объекта диагноза, связанную с тем, что проверки осуществляются в ходе его реализации. Эти методы не имеют аналогов в работах по технической диагностике. Метод последовательного безусловного выбора контрольных точек в ООП имеет аналог в технической диагностике. Однако он не просто перенесён автором на проекты – выведена формула для быстрого вычисления количества информации, получаемой при каждом сравниваемом наборе контрольных точек. Полученная формула позволяет ясно выразить, что даёт установка контрольных точек в проекте в информационном аспекте. Изначально методы были разработаны автором в стандартном предположении, что состав работ проекта неизменен. Затем результаты были распространены на мультисценарные проекты, состав работ которых может измениться в ходе реализации.

4. Методы являются эвристическими, именно поэтому предложен не один метод, а комплекс методов, совместное применение которых с последующим выбором наилучшего по предложенному автором минимаксному критерию позволяет получить вариант более близкий к оптимальному.

Перейдём к решению проблемы выбора контрольных точек в ситуации, когда нельзя пренебречь возможностью возникновения ошибок в нескольких работах проекта (кратных ошибок). Такая ситуация является характерной практически для всех проектов при выполнении работ, в которых не применяется пооперационный контроль.

2.5 Метод определения набора контрольных точек с учётом возможности наличия ошибок в нескольких работах проекта

В данном параграфе приведен разработанный автором метод выбора контрольных точек в ПССР, для которых нельзя пренебречь возможностью наличия ошибок в нескольких работах проекта. Метод опубликован в соответствующей работе [Maron M. A., 2018].

Математическая постановка задачи выбора контрольных точек в проекте может быть сформулирована следующим образом. Имеется ПССР, состоящий из $n+1$ работ. Граф сетевого графика проекта правильно пронумерован. В графе одна конечная вершина. Её номер – $n+1$. Остальные вершины соответствуют промежуточным работам. Длительность работы i равна t_i. При выполнении работ могут возникать ошибки. Причём возможно неправильное выполнение нескольких работ. Проект является типовым. Вероятность правильного выполнения проекта равна P_0.

После выполнения последней работы обязательно выполняется полная проверка. Если будет обнаружено, что ошибки имели место, то в соответствии с тем, что это ПССР, придётся переделать не только те работы, которые изначально были выполнены неправильно, но и все работы, которые используют их результаты, то есть переделать придётся все работы проекта. В результате проект будет задержан на время восстановления правильности выполнения. Далее будем называть его также «временем задержки окончания

проекта». Срыв сроков проекта вызывает финансовые потери. Эти потери линейно зависят от времени задержки окончания проекта. Время задержки окончания проекта является случайной величиной. Её значения будут различными при различных наборах ошибок. Для уменьшения этого времени можно осуществлять промежуточные полные проверки после окончания работ. С их помощью ошибки могут быть определены ранее, чем при окончательной проверке результата проекта. Длительность полной проверки после работы j равна T_j.

Требуется определить, сколько контрольных точек (промежуточных проверок) выбрать и после каких работ.

В качестве базиса решения задачи может быть использована теория информации К. Шеннона. Принципиальное отличие от особо ответственных проектов, для которых методы выбора контрольных точек предложены в предыдущем параграфе, состоит в том, что не делается предположение о наличии ошибки в одной работе. В данном случае надо учитывать, что проект может быть выполнен без ошибок, а может содержать несколько неправильно выполненных работ. Относительно ранее рассмотренного это более общий случай. Другие отличия состоят в следующем: целевой функцией является среднее время восстановления правильности выполнения; количество контрольных точек не является изначально заданным.

Для решения этой задачи требуется следующее.

- вероятностная модель возникновения ошибок в работах проекта;
- модель процесса восстановления правильности выполнения проекта и модель проекта как объекта диагноза.

Вероятностная модель возникновения ошибок в работах проекта, позволяющая вычислять вероятности ошибок в нескольких работах,

предложена автором в работе [Марон М. А., 2016] и приведена в параграфе 2.2.

Модель процесса восстановления правильности выполнения проектов типа ПССР впервые изложена в работе автора [Марон А. И., Марон М. А., 2012] и приведена в параграфе 1.3.

Модель проекта как объекта диагноза изложена в работе автора [Maron M. A., 2016] и приведена в параграфе 2.1.

Метод решения

Предлагается следующий метод решения поставленной задачи.

1. Определить проверку, энтропия результата которой максимальна.

2. Определить, уменьшает ли её выполнение среднее время восстановления правильности выполнения проекта.

3. Если не уменьшает, то прекратить дальнейший поиск проверок, которые целесообразно выполнять. Если уменьшает, то принять её в качестве проверки, которую целесообразно выполнить, и перейти к пункту 1 с учётом ранее найденных проверок [Aleskerov F., Chistyakov V., Kalyagin V., 2010].

Для вычисления энтропии результата проверок и среднего времени восстановления правильности выполнения проекта необходимо знать вероятности возникновения ошибок в работах. Они могут быть рассчитаны на основании модели, предложенной в работе [Maron M. A., 2018].

Поясним метод на примере.

Пример

Рассмотрим проект, сетевой график которого приведен на рисунке 2.3. Работы проекта сильно связаны по результатам. Плановые длительности работ в днях указаны на рисунке. Будем предполагать, что длительность исправления работы равна её плановой длительности, а длительность каждой

проверки – 2,5 дня. Допустим, что проект является типовым и из предыдущего опыта реализации известно, что 30% подобных проектов выполняются без ошибок.

После завершения последней работы проекта проверка обязательно выполняется. Это учтено в плане проекта.

Определим, какие промежуточные проверки целесообразно выполнить, или, другими словами, какие контрольные точки целесообразно запланировать для данного проекта с тем, чтобы сократить среднее время задержки завершения проекта. Причиной задержки является возможность наличия ошибок в работах проекта. Для этого применим предложенный метод.

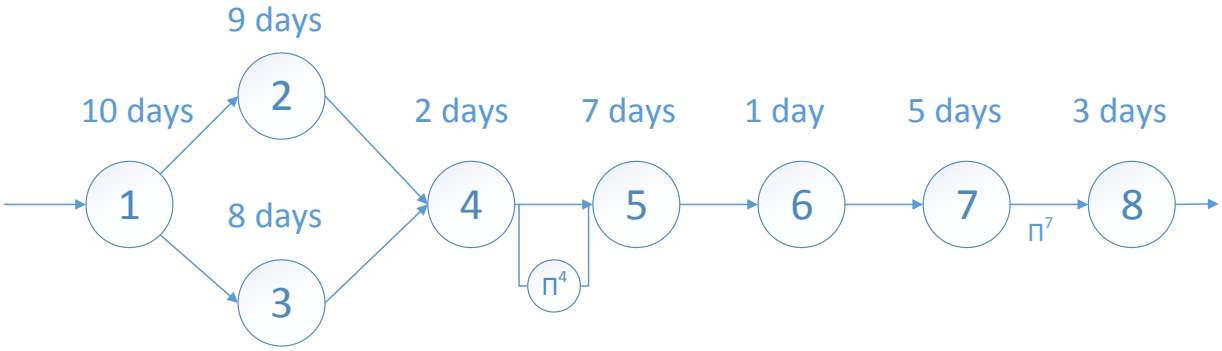

Рис. 2.3. Пример выбора контрольных точек в проекте

Начнём с вычисления вероятностей различных состояний проекта как объекта диагноза. В работе [Марон М. А., 2018] предложена модель возникновения ошибок в работах проекта. В соответствии с этой моделью на выполняемые работы действует простейший поток ошибок. Интервал действия равен τ – суммарному времени выполнения всех работ – сумме t_i по i от 1 до n. Интенсивность потока ошибок (среднее число ошибок, возникающих в единицу времени) можно найти по формуле (2.4):

$$\lambda = -\frac{lnP_0}{\tau},$$

где P_0 – вероятность того, что ошибок в проекте нет.

Для рассматриваемого примера: $P_0 = 0{,}3$; $\tau = 45$ [дней]; откуда по формуле (2.4) имеем $\lambda = 0{,}027$ [1/день].

Выполним расчёт энтропии результата каждой допустимой проверки. Результаты расчёта приведены в таблице 2.1.

Таблица 2.1

Контрольная точка	$P(\pi_1^j)$	$P(\pi_0^j)$	$H(П^j)$ [нит]
$П^1$	0,765	0,235	0,545
$П^2$	0,601	0,399	0,672
$П^3$	0,618	0,382	0,665
$П^4$	0,460	0,540	0,690
$П^5$	0,382	0,618	0,665
$П^6$	0,372	0,628	0,660
$П^7$	0,325	0,675	0,631

Энтропия результата проверки $П^4$ максимальна. Именно эта проверка даст максимум информации о состоянии проекта. На качественном уровне нетрудно объяснить, почему именно проверка, результат которой имеет максимальную неопределённость, наиболее информативна. Проверка – эксперимент. Максимум информации даёт результат эксперимента, который изначально был менее всего предсказуем. В рамках теории информации Шеннона этот факт можно строго доказать.

Энтропия результата проверки $П^4$ была вычислена следующим образом. Эта проверка будет иметь положительный результат, если каждая из работ {1;2;3;4} будет выполнена правильно – без ошибки. Вероятность такого события определяется выражением:

$$P(\pi_1^4) = exp(-\lambda * (t_1 + t_2 + t_3 + t_4)) = exp(-0{,}027 * 29) \approx 0{,}460.$$

Вероятность отрицательного результата – $P(\pi_0^4)$ дополняет эту величину до единицы и равна **0,54.**

Соответственно, энтропия определяется выражением:

$H(\text{П}^4) = -P(\pi_1^4)lnP(\pi_1^4) - P(\pi_0^4)lnP(\pi_0^4) = -0{,}46ln0{,}46 - -0{,}54ln0{,}54 = 0{,}69[\text{нит}]$.

Аналогично вычисляются энтропии результатов других проверок.

Использование натурального основания логарифма и соответствующей единицы измерения количества информации выбрано исходя из того, что стоящие под знаком логарифма величины изменяются по экспоненте.

Теперь, в соответствии с пунктом 2 предложенного метода, определим, уменьшит ли выполнение проверки П^4 среднее время восстановления правильности выполнения проекта.

Возможны следующие стратегии:

A_{11} – не выполнять проверку П^4;

A_{12} – выполнить проверку П^4.

Каждой из них соответствует своё время задержки окончания проекта, которое зависит от того, какое из возможных состояний возникнет в процессе реализации проекта. Принципиально различными по времени восстановления правильности выполнения проекта являются следующие четыре состояния:

– C_{00} – ошибок не будет ни в работах {1; 2; 3; 4}, ни в работах {5; 6; 7; 8};

– C_{01} – ошибок в работах {1; 2; 3; 4} нет, а в группе работ {5; 6; 7; 8} ошибки есть;

– C_{10} – ошибки в группе {1; 2; 3; 4} работ есть, а в работах {5; 6; 7; 8} нет;

– C_{11} – ошибки есть и в группе работ {1; 2; 3; 4}, и в группе работ {5; 6; 7; 8}.

Вероятности состояний и соответствующие им значения времени восстановления правильности выполнения проекта приведены в таблице 2.2.

Таблица 2.2

Состояние	C_{00}	C_{01}	C_{10}	C_{11}	
Вероятность	0,3	0,160	0,352	0,188	
Стратегия	Время восстановления правильности выполнения проекта				Среднее
A_{11}	0	45	45	45	31,5
A_{12}	2,5	18,5	31,5	47,5	23,7

Вероятность состояния C_{00} равна изначально заданной вероятности правильного выполнения проекта. При стратегии A_{11} в этом случае задержки окончания проекта не будет. При стратегии A_{12} возникает время задержки окончания проекта на время T = 2,5 дня, которое будет потрачено на выполнение проверки $П^4$.

Вероятность состояния C_{01} равна произведению вероятностей случайных событий:

– ошибок нет в работах {1; 2; 3; 4};

– ошибки есть в группе работ {5; 6; 7; 8}.

Её можно вычислить следующим образом:

$P(C_{01}) = exp\big(-\lambda(t_1 + t_2 + t_3 + t_4)\big)\,(1 - exp\big(-\lambda(t_5 + t_6 + t_7 + t_8)\big) = 0{,}160.$

Если это состояние возникнет, то при выполнении проверки $П^4$ будет получен положительный результат. После чего будут выполнены работы {5; 6; 7; 8}. После завершения последней – восьмой работы проекта – будет выполнена проверка, которая будет иметь отрицательный результат. Соответственно, при стратегии A_{12} придётся переделать только работы: 5; 6; 7; 8. Задержка окончания проекта в этом случае будет складываться из

их суммарной длительности и времени выполнения проверки $П^4$ и составит 18,5 дня. Напомним, что проверка после работы 8 является обязательной и её время учтено в плановой длительности проекта. Если же принять стратегию A_{11} , то время задержки окончания проекта для этого состояния будет равно $\tau = 45$ [дней].

Аналогично находятся вероятности состояний и соответствующие им значения времени задержки окончания проекта при указанных стратегиях.

Стратегии A_{12} соответствует меньшее среднее время задержки окончания проекта. Следовательно, проверку $П^4$ надо выполнить.

Определим, целесообразно ли ещё выполнять промежуточные проверки.

В таблице 2.3 приведены значения энтропии результатов проверок, которые можно выполнить с учётом того, что проверка $П^4$ будет выполнена. Проверка $П^7$ имеет максимальную энтропию результата.

Таблица 2.3

Контрольная точка	$P(\pi_1^j)$	$P(\pi_0^j)$	$H(П^j)$ [нит]
$П^5$	0,829	0,171	0,457
$П^6$	0,807	0,193	0,490
$П^7$	0,706	0,294	0,605

Здесь надо обратить внимание на следующее. Наличие или отсутствие ошибок в работах {1; 2; 3; 4}, которое будет выявлено проверкой $П^4$**,** никак не влияет на вероятности возникновения ошибок при выполнении работ {5; 6; 7; 8}. Интенсивность потока ошибок не меняется! Соответственно, вероятность положительного результата проверки $П^7$ вычисляется подобно тому, как ранее была вычислена вероятность положительного результата проверки $П^4$:

$$P(\pi_1^7) = \exp(-\lambda * (t_5 + t_6 + t_7)) = \exp(-0{,}027 * 13) \approx 0{,}706.$$

Вероятность отрицательного результата – $P(\pi_0^7)$ – дополняет эту величину до единицы. Аналогично вычисляются вероятности результатов других проверок, после чего рассчитываются соответствующие им энтропии.

Теперь надо определить, уменьшит ли выполнение проверки $П^7$ среднее время задержки проекта при условии, что до неё будет выполнена проверка $П^4$.

Проверки $П^4$ и $П^7$ разбивают работы проекта на три группы: {1; 2; 3; 4}; {5; 6; 7}; {8}. Однако при выполнении проверки $П^7$ можно быть уверенным, что в работах {1; 2; 3; 4} ошибок нет. Действительно, результат проверки $П^4$ покажет, были ошибки при их выполнении или нет. Если ошибки были, то они будут устранены. Поэтому, как и ранее, достаточно рассмотреть четыре возможных состояния:

– C_{000} – ошибок нет;

– C_{001} – ошибка в работе {8};

– C_{010} – ошибки в группе {5; 6; 7};

– C_{011} – ошибки есть и в группе работ {5; 6; 7}, и в работе {8};

и две конкурирующие стратегии:

– A_{21} – не выполнять проверку $П^7$;

– A_{22} – выполнять проверку $П^7$.

Вероятности состояний и соответствующие им значения времени восстановления правильности выполнения проекта приведены в таблице 2.4.

Таблица 2.4

Состояние	C_{000}	C_{001}	C_{010}	C_{011}	
Вероятность	0,652	0,054	0,271	0,023	
Стратегия	Время восстановления правильности выполнения проекта				Среднее
A_{21}	0	16	16	16	5,6
A_{22}	2,5	5,5	15,5	18,5	6,6

Вероятность состояния C_{000} – это условная вероятность. Она показывает вероятность того, что ошибок к моменту окончания проекта не будет при условии, что среди работ {1; 2; 3; 4} ошибок нет. Она не равна P_0, и вычисляется по формуле:

$$P(C_{000}) = exp\big(-\lambda(t_5 + t_6 + t_7 + t_8)\big) = 0{,}652.$$

Как видно из данных, приведенных в таблице 2.4, выполнение проверки Π^7 не уменьшит среднее время задержки проекта. Поэтому делаем вывод, что при реализации данного проекта следует выполнить одну промежуточную проверку – Π^4. Задача решена.

Запишем теперь метод в общем виде.

На первом шаге из множества возможных проверок $\{\Pi^j\}$ выбирается проверка Π^{r_1}, энтропия результата которой максимальна:

$$\Pi^{r_1} \rightarrow$$

$$max_{1\le j\le n} H\big(\pi^j\big) = max_{1\le j\le n}\{ exp(-\lambda(\textstyle\sum_{i\in G\left(\pi_0^j\right)} t_i))\, ln\exp(-\lambda(\textstyle\sum_{i\in G\left(\pi_0^j\right)} t_i)) -$$

$$(1 - exp(-\lambda(\textstyle\sum_{i\in G\left(\pi_0^j\right)} t_i))\, ln(1 - \exp(-\lambda(\textstyle\sum_{i\in G\left(\pi_0^j\right)} t_i)), \qquad (2.30)$$

где $G(\pi_0^j)$ – подмножество работ, при которых проверка Π^j имеет отрицательный результат;

t_i – длительность работы i.

Теперь надо определить, уменьшает ли среднее время восстановления правильности выполнения проекта осуществление проверки Π^{r_1}.

Альтернатива: A_{11} – не выполнить проверку Π^{r_1}; A_{12} – выполнять Π^{r_1}. Выбор первого варианта означает, что будет выполнена только обязательная проверка Π^{n+1} после последней работы проекта.

Подсчитаем среднее время восстановления для указанных вариантов. Проверка $\pi_1^{r_1}$ разбивает множество работ проекта на подмножества: $G(\pi_0^{r_1})$ и

$G(\pi_1^{r_1})$, соответствующие её отрицательному и положительному результатам. При реализации проекта могут возникнуть следующие 4 состояния:

– C_{00} – ошибок нет ни в $G(\pi_0^{r_1})$, ни в $G(\pi_0^{r_1})$;

– C_{01} ошибки только в работах, входящих в $G(\pi_0^{r_1})$;

– C_{10} – ошибки только в работах, входящих в $G(\pi_1^{r_1})$;

– C_{11} – ошибки и в $G(\pi_0^{r_1})$, и в $G(\pi_1^{r_1})$.

Вероятности этих состояний:

$$P(C_{00}) = exp(-\lambda(\sum_{i\in G(\pi_0^{r_1})} t_i + \sum_{i\in G(\pi_1^{r_1})} t_i)) = exp(-\lambda \sum_{G_0} t_i),$$

$$P(C_{01}) = exp(-\lambda \sum_{i\in G(\pi_0^{r_1})} t_i) * (1 - exp(-\lambda \sum_{i\in G(\pi_1^{r_1})} t_i)), \qquad (2.31)$$

$$P(C_{10}) = (1 - exp(-\lambda \sum_{i\in G(\pi_0^{r_1})} t_i)) * exp(-\lambda \sum_{i\in G(\pi_1^{r_1})} t_i),$$

$$P(C_{11}) = (1 - exp(-\lambda \sum_{i\in G(\pi_0^{r_1})} t_i)) * (1 - exp(-\lambda \sum_{i\in G(\pi_1^{r_1})} t_i)).$$

Заметим, что на первом шаге вероятность $P(C_{00}) = P_0$.

Время восстановления правильности выполнения проекта при возникновении состояния C_{00} будет равно:

$T(C_{00}, A_{11}) = 0$ – для варианта A_{11};

$T(C_{00}, A_{12}) = T_{r_1}$ – для варианта A_{12}.

Время восстановления правильности выполнения проекта при возникновении состояния C_{01}:

– $T(C_{01}, A_{11})$ –для варианта A_{11} будет равно сумме длительностей работ, входящих в $\mathrm{G}(\pi_1^{r_1})$ и $\mathrm{G}(\pi_0^{r_1})$, то есть сумме длительностей всех работ проекта $i \in G_0$;

– $T(C_{01}, A_{12})$ – для варианта A_{12} будет равно T_{r_1} плюс сумма длительностей работ, входящих в $\mathrm{G}\left(\pi_1^{r_1}\right)$.

Время восстановления правильности выполнения проекта при возникновении состояния C_{10}:

– $T(C_{10}, A_{11})$ – для варианта A_{11} будет равно сумме длительностей всех работ проекта $i \in G_0$;

– $T(C_{10}, A_{11})$ – для варианта A_{12} будет равно T_{r_1} плюс сумма длительностей всех работ, входящих в $\mathrm{G}\left(\pi_0^{r_1}\right)$.

Время восстановления правильности выполнения проекта при возникновении состояния C_{11}:

– $T(C_{11}, A_{11})$ – для варианта A_{11} будет равно сумме длительностей всех работ проекта $i \in G_0$;

– $T(C_{11}, A_{12})$ – для варианта A_{12} будет равно T_{r_1} плюс сумме длительностей всех работ проекта $i \in G_0$.

Вычислим значения:

$$T(A_{11}) = P(C_{00})T(C_{00}, A_{11}) + P(C_{01})T(C_{01}, A_{11}) + P(C_{10})T(C_{10}, A_{11}) + P(C_{11})T(C_{11}, A_{11})$$

$$T(A_{12}) = P(C_{00})T(C_{00}, A_{12}) + P(C_{01})T(C_{01}, A_{12}) + P(C_{10})T(C_{10}, A_{12}) + P(C_{11})T(C_{11}, A_{11}) \quad (2.32)$$

Если не выполняется неравенство:

$$T(A_{11}) < T(A_{12}), \quad (2.33)$$

то выполнение найденной проверки считаем нецелесообразным и никаких контрольных точек больше не выбираем.

Если неравенство (2.23) выполняется, то выполняем следующие действия.

Принимаем Π^{r_1} в качестве первой контрольной точки в проекте.

Продолжаем выбор контрольных точек среди подмножества работ $G(\pi_1^{r_1})$.

Для полной аналогии с ранее выполненными действиями полагаем, что:

$G(\pi_1^{r_1}) = G_0$, то есть удаляем из дальнейшего рассмотрения все работы, входящие в $G(\pi_0^{r_1})$.

Важно отметить, что в данном случае вероятности ошибок в работах проекта на всех этапах выбора контрольных точек не зависят от того, какие проверки выполнены ранее. Интенсивность потока ошибок не меняется!

Таким образом, в данном параграфе изложен следующий новый научный результат.

1. Впервые предложен метод определения количества и мест расположения контрольных точек, который можно применить к большинству типовых проектов. В нём учитывается как возможность наличия ошибок в нескольких работах, так и затраты времени на выполнение операций контроля. Метод не имеет аналогов в работах по технической диагностике.

2.6 Выводы по главе 2

В настоящей главе приведено решение следующих задач диссертационного исследования, поставленных в главе 1 на основании анализ проработанности проблемы диагностики проектов:

1. Разработать диагностическую модель проекта. Эта модель должна давать возможность установить связь между ошибками в работах и результатами проверок.

2. Разработать математическую модель возникновения ошибок в работах, которая позволит рассчитывать вероятности ошибок.

3. Предложить критерии для сравнения вариантов расстановки контрольных точек в зависимости от того, как потери от задержки окончания проекта, вызванной ошибками в работах, зависят от времени восстановления правильности его выполнения.

4. Разработать методы определения наборов контрольных точек в соответствии с предложенными критериями и экспериментально оценить их эффективность, если это будут эвристические, а не регулярные методы, гарантирующие нахождение оптимума.

При этом получены следующие новые научные результаты.

1. Впервые предложена диагностическая модель проекта (ДМП). Она позволяет представить соответствие между результатами проверок и возможными ошибками в работах в виде таблицы возможных ошибок (ТВО) проекта, для построения которой можно использовать методы, разработанные в теории графов. Используя ДМП, можно легко решить задачу определения минимального полного набора проверок, необходимого и достаточного для контроля правильности выполнения проекта. Она является основой для решения поставленной задачи определения наборов контрольных точек.

2. Впервые разработана математическая модель возникновения ошибок в работах проекта. Модель соответствует реальностям процесса возникновения ошибок, если: ошибки возникают независимо друг от друга; при реализации проекта не допускается ослабления контроля качества выполнения работ даже по мере приближения запланированных сроков его окончания; к недобросовестным исполнителям своевременно применяются меры воздействия, заставляющие их соблюдать установленные требования; нет оснований предполагать, что одни исполнители работают лучше других.

С помощью предложенной модели можно рассчитать вероятности не только для одиночных ошибок, но и для любой их комбинации. Предложена и модель расчёта вероятностей ошибок для ООП. В этом частном, но исключительно важно случае, выбор варианта расстановки контрольных точек уместно проводить в предположении о наличии ошибки в одной работе проекта.

3. Впервые предложен критерий для сравнения наборов контрольных точек в ООП для случая, когда потери от задержки проекта нелинейно зависят от времени восстановления правильности выполнения проекта.

4. Впервые предложены методы выбора контрольных точек в ООП. Методы используют предложенные автором диагностическую модель проекта (ДМП) и модель для расчёта вероятностей ошибок в ООП. Методы не являются регулярными, но их научной базой является теория информации К. Шеннона, применение которой даёт хорошие результаты при планировании экспериментов, а проверки в контрольных точках являются именно экспериментами, выполняемыми в целях диагностирования проекта.

5. Метод последовательного условного выбора контрольных точек в ООП и модифицированный метод последовательного условного выбора контрольных точек в ООП существенно используют особенность проекта как объекта диагноза, связанную с тем, что проверки осуществляются в ходе его реализации. Эти методы не имеют аналогов в работах по технической диагностике. Метод последовательного безусловного выбора контрольных точек в ООП имеет аналог в технической диагностике. Однако он не просто перенесён автором на проекты – выведена формула для быстрого вычисления количества информации, получаемой при каждом сравниваемом наборе контрольных точек. Полученная формула позволяет ясно выразить, что даёт установка контрольных точек в проекте в информационном аспекте. Изначально методы были разработаны автором в стандартном

предположении, что состав работ проекта неизменен. Затем результаты были распространены на мультисценарные проекты, состав работ которых может измениться в ходе реализации.

6. Методы являются эвристическими, именно поэтому предложен не один метод, а комплекс методов, совместное применение которых с последующим выбором наилучшего по предложенному автором минимаксному критерию позволяет получить вариант, более близкий к оптимальному.

7. Впервые предложен метод определения количества и мест расположения контрольных точек, который можно применить к большинству типовых проектов. В нём учитывается как возможность наличия ошибок в нескольких работах, так и затраты времени на выполнение операций контроля. Метод не имеет аналогов в работах по технической диагностике. Он опирается на впервые предложенные автором математические модели:

- проекта как объекта диагноза;
- возникновения ошибок в работах проекта.

Глава 3 Реализация и оценка эффективности предложенных методов выбора контрольных точек в проектах

3.1 Требования к программе оценки эффективности методов выбора контрольных точек в проектах

Теоретической базой предложенных методов выбора контрольных точек в проекте является теория информации К. Шеннона. Основанием для её применения является тот факт, что проверки, выполняемые в контрольных точках, – это эксперименты, которые должны давать как можно больше информации об исследуемом объекте. Бессмысленно выполнять проверки, результат которых заведомо известен.

Несмотря на известную эффективность информационного подхода, предложенные методы не являются регулярными – не гарантируют получения оптимальной расстановки контрольных точек. Их эффективность требует экспериментального подтверждения.

Определим, какие важнейшие параметры надо учитывать в таком эксперименте, который будем проводить методом Монте-Карло, используя рекомендации, приведенные в работах [Акопов А. С., 2014; Зажигаев Л. С., Кишьян А. А., Романников Ю. И., 1978; Бараш Л. Ю., Щур Л. Н., 2012].
Начнём с параметров особо ответственных проектов без сценариев.

Каждый проект имеет определённое количество работ и определённую логическую структуру, которую отражает сетевой граф. Каждая работа проекта имеет определённое число последователей. Соответственно, из каждой вершины сетевого графа выходит определённое число дуг. Их общее количество в проекте отражает его логическую сложность. Примем в качестве показателя логической сложности проекта величину z, равную отношению числа дуг в его сетевом графике к числу работ. Число дуг на

сетевом графике проекта равно сумме всех элементов матрицы смежности вершин графа проекта:

$$z = \frac{\sum_{n}^{i=1}\sum_{n}^{j=1} a_{ij}}{n} \quad (3.1)$$

где

n – количество вершин графа проекта, равное числу работ;

a_{ij} – элемент матрицы смежности графа проекта, при этом $a_{ij} = 1$, если есть дуга из вершины i в вершину j, и равен нулю в противном случае.

Каждая работа проекта имеет определённую плановую длительность. Если следовать рекомендациям по планированию проекта, то диапазон изменения этой величины не так уж и велик. Рекомендуется иметь в плане работы длительностью 5–7 дней. Более длинные следует разбить на составляющие. Конечно, это не всегда возможно, но в эксперименте такими случаями можно пренебречь. Соответственно, длительность работ можно охарактеризовать средним значением – $t_{\text{ср}}$ и среднеквадратичным отклонением – δ, считая, что отклонение от среднего будет в пределах $\pm 3\delta$.

Таким образом, варьируемыми параметрами проектов будут:

1) n – количество работ проекта;
2) z – показатель логической сложности проекта;
3) $t_{\text{ср}}$ – средняя плановая длительность работ;
4) δ – СКО плановой длительности.

Другой важнейший параметр, который надо задавать в экспериментах по оценке методов расстановки контрольных точек в ООП, это m – число контрольных точек (проверок).

В параграфе 2.3 изложены три метода определения наборов контрольных точек в проектах, которые здесь и в дальнейшем будем сокращенно называть:

1) МПУВ_КТ – метод последовательного условного выбора контрольных точек, он изложен в разделе 2.4.1;

2) мод. МПУВ_КТ – модифицированный метод последовательного условного выбора контрольных точек, он изложен в разделе 2.4.2;

3) МБВ_КТ – метод безусловного выбора контрольных точек, он изложен в разделе 2.4.3.

Все они должны быть реализованы в создаваемой программе.

Для оценки эффективности эвристического метода выбора контрольных точек в проекте следует сравнить значение целевой функции (2.9), соответствующее полученному набору контрольных точек, с её минимально возможным значением для данного проекта при заданном количестве контрольных точек, другими словами, со значением целевой функции, достигаемым при оптимальном наборе. Найти оптимальный набор можно, только осуществив перебор всех возможных вариантов расстановки. Соответственно, в программе должен быть реализован метод полного перебора для нахождения оптимального варианта расстановки контрольных точек и вычисления соответствующего ему минимального значения целевой функции.

Необходимость нахождения оптимального варианта полным перебором накладывает жёсткие ограничения на размерность проекта n и число контрольных точек m, для которых можно осуществить планируемый эксперимент, особенно с учётом того, что для достижения значимых результатов сравнения число прогонов должно быть велико.

В качестве показателя эффективности расстановки контрольных точек каждым из предложенных методов при одном прогоне примем величину d, равную, выраженное в процентах, относительное отклонение значения целевой функции (2.9), соответствующей найденному набору контрольных точек, от минимального значения, соответствующего оптимальному набору S^*:

$$d = \frac{T(S) - T(S^*)}{T(S^*)} * 100\% \qquad (3.2)$$

где

$T(S)$ – максимальное время восстановления правильности выполнения проекта при найденном, с помощью исследуемого метода, наборе контрольных точек – S;

$T(S^*)$ – максимальное время восстановления правильности выполнения проекта при оптимальном наборе контрольных точек, найденном полным перебором.

Программа должна обеспечить возможность статистического анализа данного показателя в эксперименте, состоящем в большом числе прогонов.

Автором были разработаны три эвристических метода выбора контрольных точек (параграф 2.3) для их практического применения следующим образом. Для заданного проекта определяются три набора контрольных точек – по одному каждым методом, и среди них выбирается лучший, которому соответствует наименьшее максимальное время восстановления правильности выполнения проекта. Соответственно, к программе предъявляется требование реализации такого комплексного подхода к выбору контрольных точек в проектах. Естественно, комплексный подход оправдан, если среди предложенных методов нет метода, который всегда лучше других. Было выдвинуто предположение, что такого метода нет. Оно полностью подтвердилось в ходе проведенного эксперимента (см. параграф 3. 3).

3.2 Описание программы оценки эффективности методов выбора контрольных точек в проектах

В соответствии с требованиями, указанными выше, был разработан программный комплекс для проведения эксперимента по оценке эффективности предложенных методов выбора контрольных точек. В дополнение к возможности проведения экспериментов в него были сразу добавлены компоненты, позволяющие осуществлять выбор контрольных точек в реальных проектах. Пока такая работа выполняется в режиме консалтинга. В дальнейшем возможно и создание коммерческого продукта для конечного пользователя – руководителя проекта, осуществляющего управление проектом с применением MS – Project. Ориентация на MS – Project вызвана тем, что он превосходит другие CASE – средства управления проектами по PMBoK по соответствующим критериям: качество реализации; наличие методического обеспечения; наличие системы обучения; наличие поддержки в России [Богданов В., 2016; Ципес Г. Л., Товб А. С., 2009].

Соответственно, в качестве среды разработки была выбрана программа от того же производителя Microsoft Visual Studio, так как она имеет встроенную библиотеку, обеспечивающую возможность подключения к файлам с расширением .mpp из программ, разработанных в данной среде разработки [Рихтер Д., 2017; Andrew T., 2012; Perkins B., Hammer J., Reid J., 2018; Iain D., 2001; Johnson B., 2017; Макконнелл С., 2010]. При разработке использованы рекомендации, приведенные в книгах [Крук Е. А., Овчинников А., 2014; Кнут Д., 2017].

Интерфейс программы приведен на рисунках 3.1, 3.2, 3.3, 3.4.

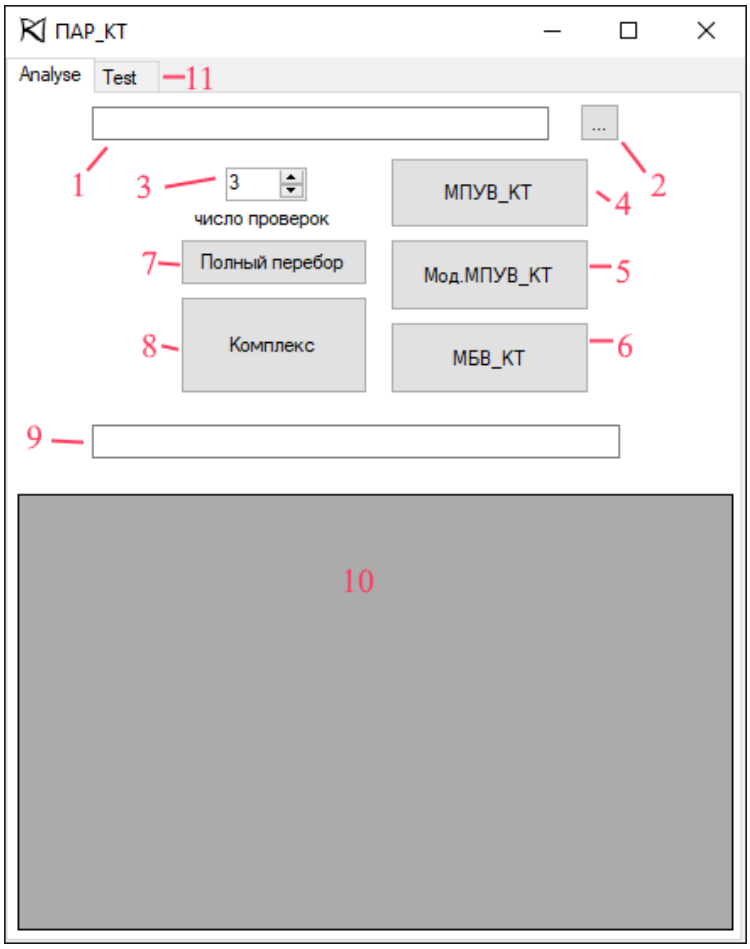

Рис. 3.1. Интерфейс программы. Расчет для проекта

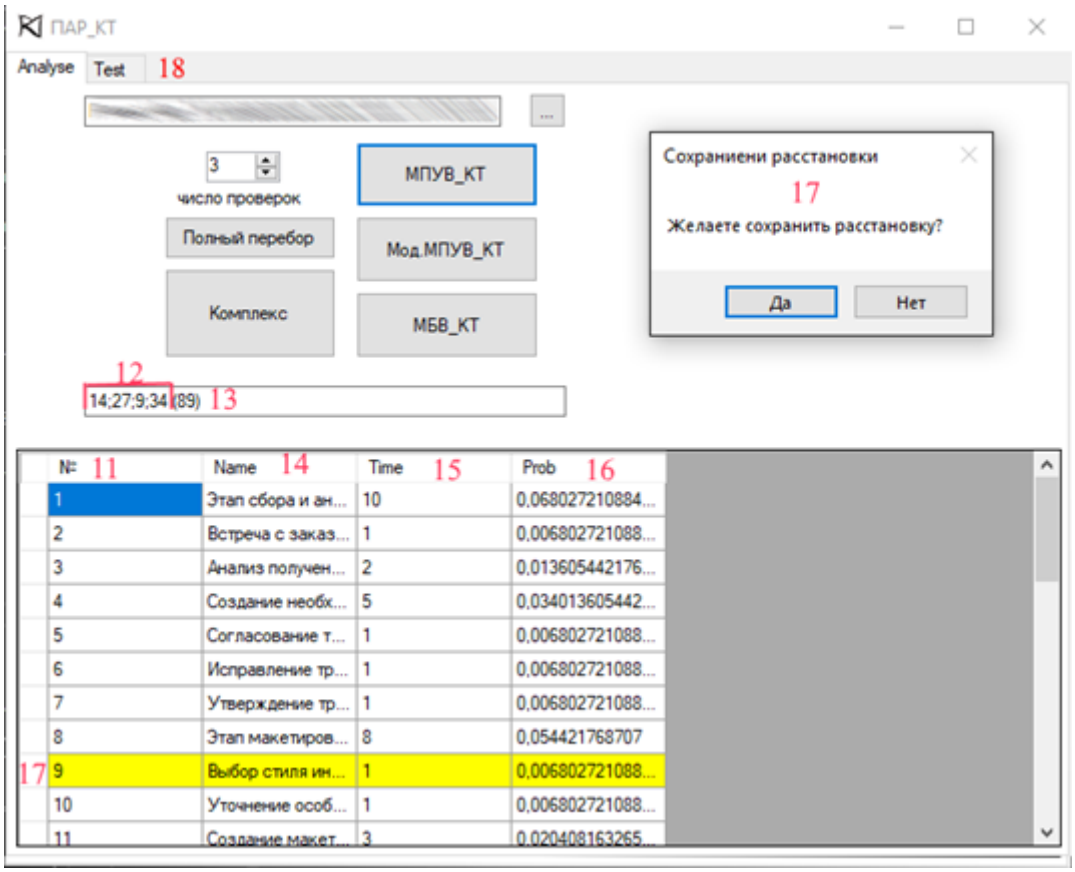

Рис. 3.2. Интерфейс программы. Пример результата расчетов

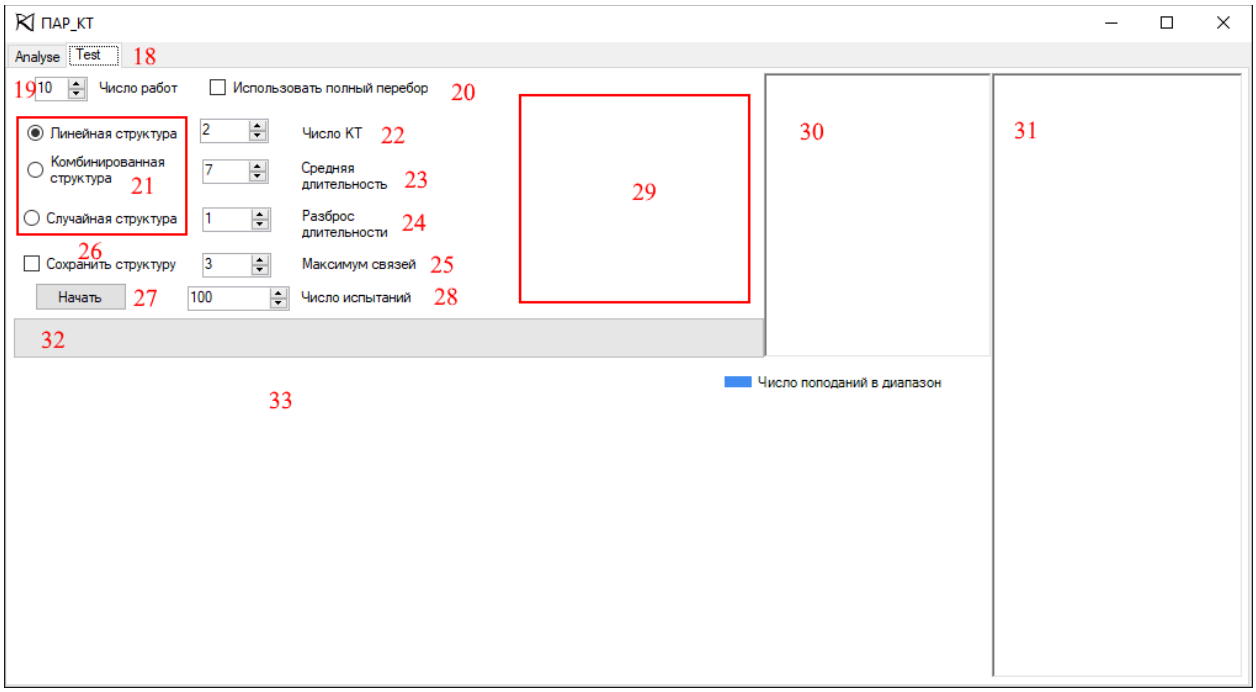

Рис. 3.3. Интерфейс программы. Ввод параметров расчёта эксперимента

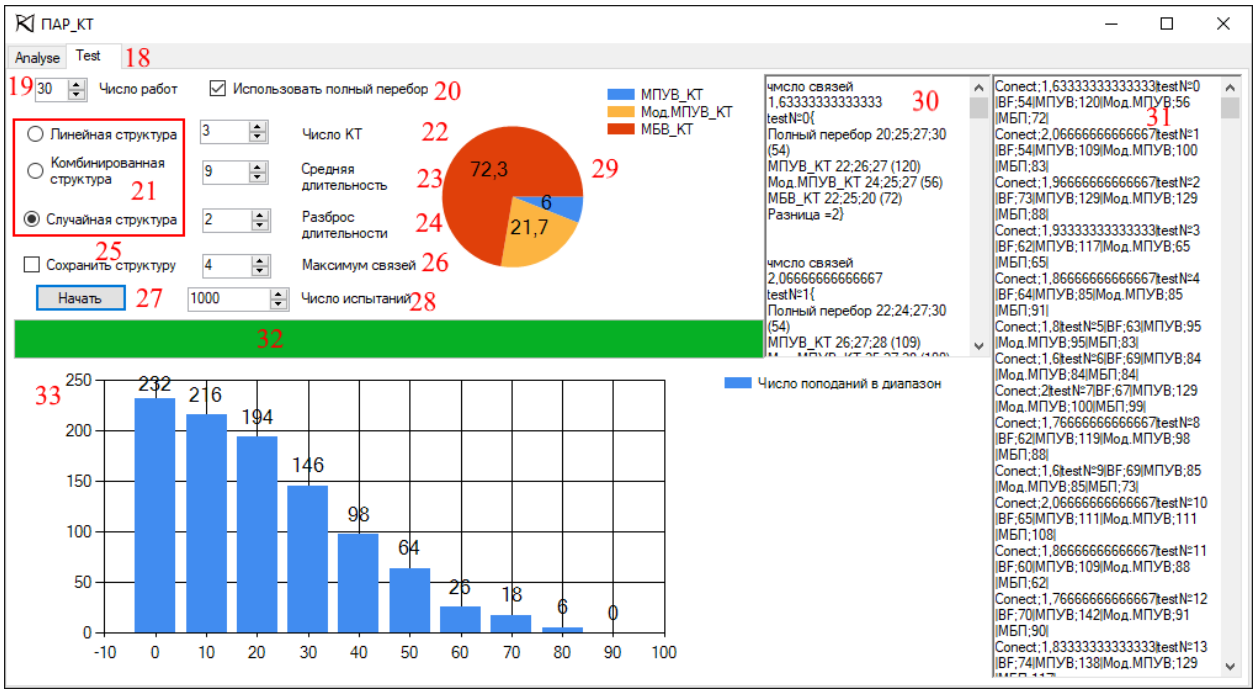

Рис. 3.4. Интерфейс программы. Пример результата расчёта

На рисунках 3.1, 3.2, 3.3, 3.4 применена следующая нумерация элементов интерфейса.

1. Строка имени файла проекта, загружаемого в программу в формате mpp.
2. Кнопка показа диалогового окна открытия файла.
3. Элемент «вверх–вниз» для указания числа контрольных точек в проекте.
4. Кнопка вызова метода последовательного условного выбора контрольных точек.
5. Кнопка вызова модифицированного метода последовательного условного выбора контрольных точек.
6. Кнопка вызова метода безусловного выбора контрольных точек.
7. Кнопка вызова метода полного перебора.
8. Кнопка вызова «Комплекса».
9. Окно выдачи результатов мест размещения контрольных точек и значения критерия.
10. Окно вывода дополнительных характеристик проекта.

11. Порядковый номер работы.

12. Номера работ, где необходимо установить контрольные точки, включая последнюю работу, где контрольная точка ставится всегда.

13. Значение критерия.

14. Название работы.

15. Длительность работы.

16. Вероятность появления ошибки в данной работе.

17. Дополнительная визуализация мест расстановки контрольных точек.

18. Выбор окна работы с заданным проектом или проведение эксперимента.

19. Число работ в генерирующихся для эксперимента проектах.

20. Использование полного перебора.

21. Логические структуры генерирующихся проектов.

22. Число контрольных точек в проектах.

23. Средняя длительность работ проекта.

24. СКО длительности работ (σ), соответственно $t_{\text{мин}} = t_{\text{ср}} - 3 * \sigma$, а $t_{\text{макс}} = t_{\text{ср}} + 3\sigma$.

25. Элемент управления, позволяющий сохранять сгенерированные проекты.

26. Максимальное число последователей работы.

27. Кнопка запуска расчётов.

28. Число испытаний (прогонов) в эксперименте.

29. График процентного числа наилучших результатов для каждого метода.

30. Окно вывода каждого конкретного испытания.

31. Окно вывода дополнительных сведений о сгенерированном проекте, если число прогонов 1.

32. Строка прогресса, показывающая текущее состояние расчетов

33. График числа попаданий значения d в указанные диапазоны.

Дадим необходимые пояснения и уточнения.

Для защиты от зацикливания установлены ограничения, связанные с полным перебором. Здесь приходится искать компромисс между целесообразностью проведения эксперимента для проектов с не слишком малым числом работ и практической реализуемостью эксперимента. При каждом прогоне число вариантов расстановок контрольных точек определяется количеством сочетаний из $n-1$ (как и ранее, рассматриваются проекты с одной конечной работой, после которой проверка всегда выполняется) по m. Если m близко к $n/2$, то с ростом n число вариантов стремительно растёт. Для большинства проектов возможным является выделение денег на проведение такого числа тотальных проверок, которое соответствует от 5% до 15% от общего числа работ в проекте. Поэтому принято решение в экспериментах выбирать количество контрольных точек на уровне 10% от числа работ.

Эксперимент должен быть реализуем минимум для 500 прогонов. Тогда в нашем случае эксперимент можно считать представительным, и не прибегая к более тонкому определению необходимого объёма выборки [Марон М. А., 2010]. С учётом этого, как показала практика расчётов, они выполнимы за приемлемое время при числе работ в проекте не большем, чем 40 [Воскобойников Ю., 2016].

В программе вероятности ошибок в работах рассчитываются на основании длительности работ по формуле (2.7).

Программа позволяет не только принять проект из MS-Project, но и записать в MS-Project сгенерированный проект.

Здесь «Линейная структура» – это последовательная цепочка работ. «Комбинированная структура» – это проект типа, приведенного на рисунке 3.5.

Выбор опции «Случайная структура» является основным при проведении экспериментов. Программа создаёт сетевой график проекта, генерирующий случайным образом для каждой вершины количество последователей и их номера с учётом заданного максимального количества последователей, и не нарушая принцип правильной нумерации графа. Также случайным образом определяется длительность работы с учётом заданных значений средней длительности и СКО.

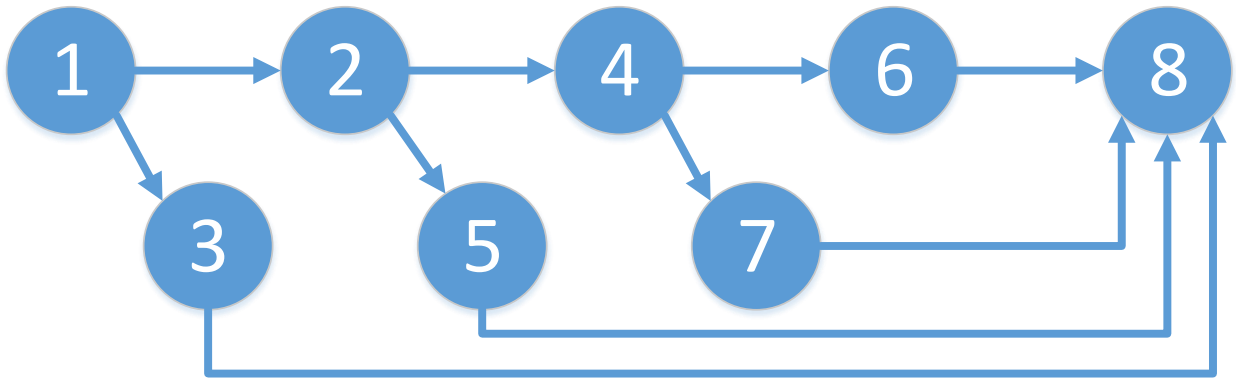

Рис. 3.5. Пример графа проекта типа «комбинированная структура»

Для сгенерированного графа рассчитывается по формуле (3.1) полученный показатель логической сложности z. Минимальное значение величины z равно $1-\frac{1}{n}$, которое достигается в проектах, имеющих линейную структуру. Максимальное же значение данного параметра не может превышать $\frac{(n-1)}{2}$.

Выдержки из текста программы приведены ниже.

Для подключения к MS-Project была использована стандартная библиотека Microsoft.Office.Interop.MSProject [Михеев Р.,2006]. Использование данных из файла MS-Project осложняется тем, что данные можно получить только в виде текста, кроме того, длительность работ хранится в единицах, равных 3 минутам.

Подключение к файлу происходит следующим образом, показанным в коде ниже:

```
a = new Microsoft.Office.Interop.MSProject.Application();// Открытие MS-Project
            proj = a.Projects.Add(false, textBox1.Text, false); //Открытие файла
выбранного пользователем
            tasksOfProj = tasksToStrMas(proj.Tasks);    //Получение данных из файла
проекта
            conmas = fill_conmas(tasksOfProj, 1);//Получение таблицы смежности из
информации о предшественниках
            desttbl = fill_destmas(conmas); //Получение таблицы связности из таблицы
смежности

            for (int i = 0; i < tasksOfProj.Length; i++) //Заполнение массива
длительностей в днях
            {
                durmas[i] = double.Parse(tasksOfProj[i][1]) / 480;
            }
            probmas = Calcprob(durmas); //заполнение массива вероятностей
```

Сама процедура получения данных из файла выглядит следующим образом, показанным в коде ниже:

```
public string[][] tasksToStrMas(Tasks t)
        {
            string[][] taskmas = new string[t.Count][];//Массив работ
            string[] pred;//Массив предшественников

            int i = 0;//Номер работы

            foreach (Microsoft.Office.Interop.MSProject.Task ts in t) //Перебор каждой
работы в проекте
            {
                pred = ts.Predecessors.Split(';'); //Получение массива предшественников

                taskmas[i] = new string[2 + pred.Length]; //Создание массива данных для
каждой работы
                taskmas[i][0] = ts.Name;//Запись имени работы
                taskmas[i][1] = "" + ts.Duration;//Запись длительности работы
                for (int j = 0; j < pred.Length; j++)//запись предшественников
                {
                    if (pred[j] != "")
                        taskmas[i][j + 2] = pred[j];
                    else
                        taskmas[i][j + 2] = "" + ts.ID;
                }
                i++;
            }

            return taskmas;
        }
```

Процедура получения матрицы связности и матрицы смежности:

```
  public int[,] fill_conmas(string[][] tabl, int type)
        {
            int[,] rez = new int[tabl.Length, tabl.Length];//Создание пустой матрицы
смежности
```

```
            for (int i = 0; i < tabl.Length; i++)
            {
                for (int j = 1 + type; j < tabl[i].Length; j++)
                {

                    rez[i, int.Parse(tabl[i][j]) - 1] = 1;//Преобразование

                }
                rez[i, i] = 1;
            }

            return rez; //Возвращение результата

        }
        public int[,] fill_destmas(int[,] con)
        {
            int[,] rez = new int[con.GetLength(1), con.GetLength(1)];//Создание пустой
матрицы
            for (int i = 0; i < con.GetLength(1); i++)//Заполнение созданной матрицы
данными из матрицы смежности
                for (int j = 0; j < con.GetLength(1); j++)
                    rez[j, i] = con[i, j];

            for (int k = 0; k < rez.GetLength(1); k++)//Алгоритм Уоршалла для получения
матрицы связности
                for (int i = 0; i < rez.GetLength(1); i++)
                    for (int j = 0; j < rez.GetLength(1); j++)
                        rez[i, j] = Math.Max(rez[i, j], (rez[i, k] + rez[k, j]) / 2);
            return rez; //Возвращение результата

        }
```

Процедура расстановки КТ методом ПУВ:

```
public string PlaceKT_OW(int[,] destmas, double[] durmas, int numberKT)//
Последовательный условный выбр КТ
        {
            double[] prob = Calcprob(durmas);//Создание матрицы вероятностей и пересчет
вероятностей
            double max = CalcEnt(destmas, prob, 0),//Максимум энтропии с расчетом ее
значения для 0 работы
                cur = 0;//Значение энтропии при установке текущей точки
            string rez,//Результат расстановки
                pt;//Расстановка оставшихся точек
            int place = -1;//Текущее место расстановки

            for (int i = 1; i < destmas.GetLength(0); i++)//Установка текущей точки
            {
                cur = CalcEnt(destmas, prob, i);
                if (cur > max)
                {
                    max = cur;
                    place = i;
                }
            }

            rez = "" + place + ";";//Запись текущей точки

            if (numberKT > 1 && place != -1)//Расстановка
            {
                pt = PlaceKT_OW(CutDestmas(destmas, place), CutMas(destmas, durmas,
place), numberKT - 1);//Рекурсивный вызов процедуры расстановки числа точек - 1
```

```
            if (pt != "-1;")//Установка точки если алгоритм не дошел до конца
                rez += pt;
        }

        return rez;//Возвращение результата
    }
```

Процедура расстановки КТ методом Мод.МПУВ:

```
public string PlaceKT_MOW(int[,] destmas, double[] durmas, int
numberKT)//Модифицированный последовательный условный выбор
    {
        double[] prob = Calcprob(durmas);//Создание матрицы вероятностей и пересчет
вероятностей
        double max = CalcR(destmas, prob, 0, durmas), //Максимум энтрапии с расчетом
ее значения для 0 работы
            cur = 0;//Значение энтропии при установке текущей точки
        string rez, //Результат расстановки
            pt;//Расстановка оставшихся точек
        int place = -1;//Текущее место расстановки

        for (int i = 1; i < destmas.GetLength(0); i++)//Установка текущей точки
        {
            cur = CalcR(destmas, prob, i, durmas);
            if (cur > max)
            {
                max = cur;
                place = i;
            }
        }

        rez = "" + place + ";";//Запись текущей точки
        if (numberKT > 1 && place != -1)//Расстановка оставшихся точек
        {
            pt = PlaceKT_OW(CutDestmas(destmas, place), CutMas(destmas, durmas,
place), numberKT - 1);//Рекурсивный вызов процедуры расстановки числа точек - 1
            if (pt != "-1;")//Установка точки, если алгоритм не дошел до конца
                rez += pt;
        }


        return rez;//Возвращение результата
    }
```

Процедура расстановки КТ методом МБВ:

```
public string PlaceKT_TW(int[,] destmas, double[] probmas, int numberKT, string placedKT,
bool isLL)//Безусловный выбор
    {

        double min = 0,//Минимальное значение
            curent = 0;//Текущее значение
        int[] placnumb = getKT(placedKT, ';', 1);//Масивв КТ
        string rez = placedKT;//Результат
        bool here = true,//Переменная, показывающая, что тут точка уже стоит
            first = true;//Переменная, показывающая, что идет растановка первой точки
        int maxnum = -1;//Место с наибольшим значением

        for (int i = 0; i < probmas.Length - 1; i++)//Установка точки
        {
```

```
                here = false;
                for (int j = 0; j < placnumb.Length; j++)//Проверка на уже расставленных
точках
                {
                    if (i == placnumbEEj])
                    {
                        here = true;
                        break;

                    }
                }
                if (!here)
                {
                    curent = calcDW(splitGroup(destmas, getKT(rez + i + ";", ';', 1)),
probmas, isLL);
                    if (first)
                    {
                        first = false;
                        min = curent;
                        maxnum = i;
                    }
                }
                if (min > curent)
                {
                    min = curent;
                    maxnum = i;
                }

            }
            rez += maxnum + ";";//Запись текущей точки

            if (numberKT > 1)//Расстановка оставшихся точек
            {
                rez = PlaceKT_TW(destmas, probmas, numberKT - 1, rez, isLL);//Рекурсивный
вызов процедуры для числа точек-1
            }

            return rez; //Возвращение результата
        }
```

Разработанная программа отвечает установленным требованиям. С её помощью проведен численный эксперимент. Его результаты приведены в следующем параграфе.

3.3 Результаты эксперимента по оценке эффективности предложенного комплекса методов выбора контрольных точек в проектах

Проведение эксперимента будет проходить в 3 этапа. На первом этапе все методы будут сравниваться отдельно для определения промежутка

значений z, где данные методы дают наилучшие результаты. Для этого будут генерироваться проекты со случайной величиной z. Затем для каждого проекта будет произведена расстановка контрольных точек каждым методом и полным перебором. Затем будет рассчитано значение d. Полученные значения будут занесены в график для каждого конкретного метода. На основании полученных графиков будут определены искомые промежутки. В результате данного этапа можно получить промежуток, на котором все методы дают наилучшие результаты. На втором этапе методы будут сравниваться между собой на найденном промежутке для обнаружения наличия или отсутствия строгого доминирования одного метода над другим или группой других. Для этого будут сгенерированы проекты в найденном на первом этапе промежутке значений z. Далее в каждом полученном проекте будут расставлены контрольные точки каждым методом. Далее для каждого результата расстановки будет посчитано и сравнено значение критерия. На основании этих данных количество ситуаций, когда метод дал строго наилучшее значений будет посчитано для каждого метода, затем нормировано и представлено в виде графика. Если будет обнаружено, что какой-либо метод не являлся лучшим ни одного раза, то данная ситуация будет основанием считать, что оставшиеся методы строго доминируют над данным. Таким образом, только доминирующие методы стоит оценивать на эффективность далее. Эти методы войдут в «комплекс». Результатом расстановки «комплекса» является лучшая расстановка из числа методов, входящих в него. На третьем этапе будут генерироваться проекты с разным числом работ и значением сложности z. В полученных проверках будет проводиться расстановка «комплексом» и полным перебором. Далее будет вычисляться отклонение от оптимальной расстановки. Данные значения будут выводиться в таблицу, что и даст основание сказать об эффективности «комплекса». Кроме того, по изменению значений d относительно

увеличения числа работ можно будет сделать вывод о степени изменения отклонений от оптимума с увеличением числа работ.

Этап 1

Метод последовательного условного выбора контрольных точек

Для проведения первого этапа испытания было сгенерировано 500 проектов с числом работ 30, числом контрольных точек 3, средней длительностью 7 д., разбросом длительностей 1 д. и со следующими значениями переменной:

- 100 проектов с линейной структурой;
- 100 проектов со случайной структурой и числом связей меньше, чем 4;
- 100 проектов со случайной структурой и числом связей меньше, чем 6;
- 100 проектов со случайной структурой и числом связей меньше, чем 8;
- 100 проектов со случайной структурой и числом связей меньше, чем 10.

После проведения расчетов были получены следующие данные, представленные на рисунках 3.6, 3.7, 3,8.

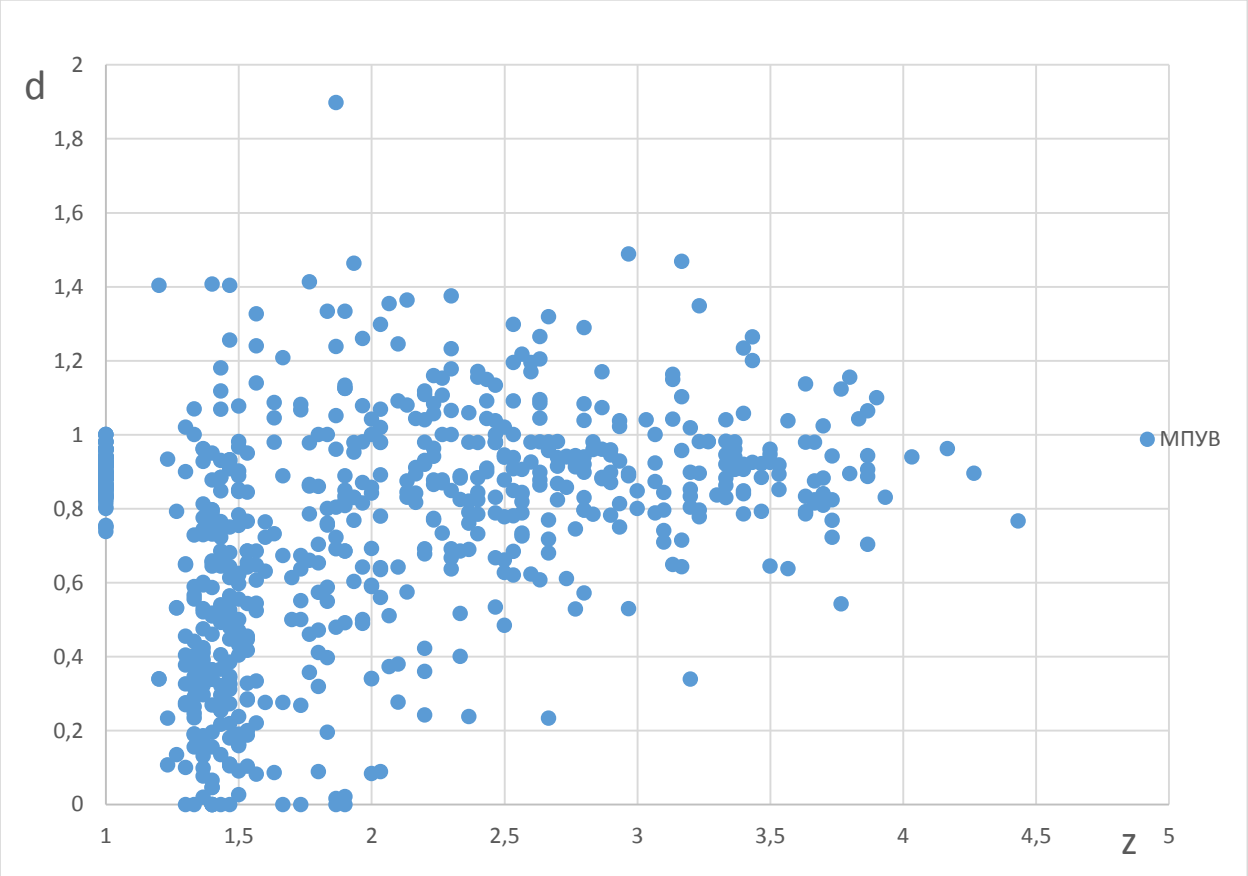

Рис. 3.6. Зависимость результата расстановки методом МПУВ от сложности проекта

На рисунке 3.6 видно, что наилучшее значение данный метод проявляет в диапазоне $1.25 < z < 1.9$, кроме того, с ростом значения $z > 2$ результаты работы метода ухудшаются.

Модифицированный метод последовательного условного выбора контрольных точек

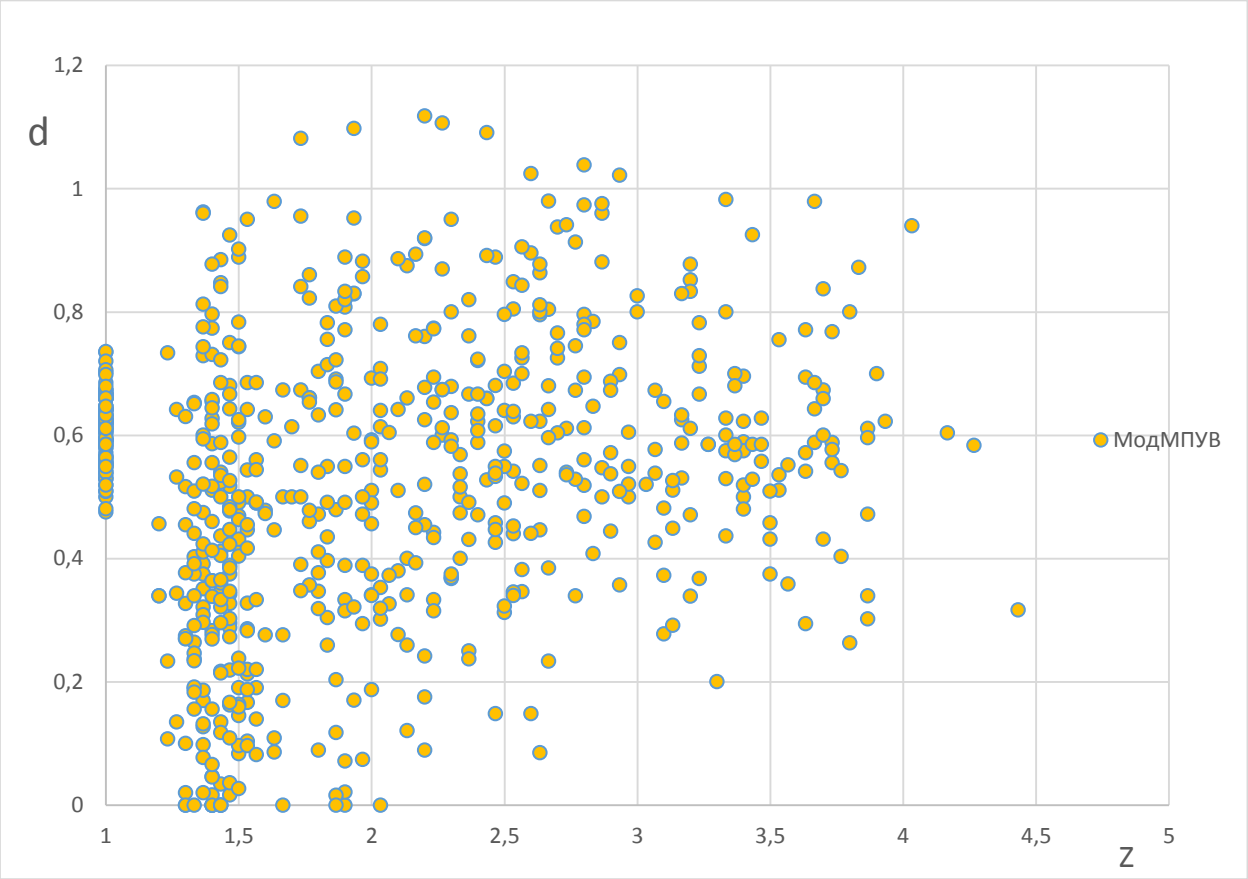

Рис. 3.7. Зависимость результата расстановки методом Мод.МПУВ от сложности проекта

На рисунке 3.7 видно сходство с рисунком 3.6 МПУВ, однако заметно, что диапазон наилучших результатов для метода Мод.МПУВ увеличивается до значения z= 2.1.

Метод безусловного выбора контрольных точек

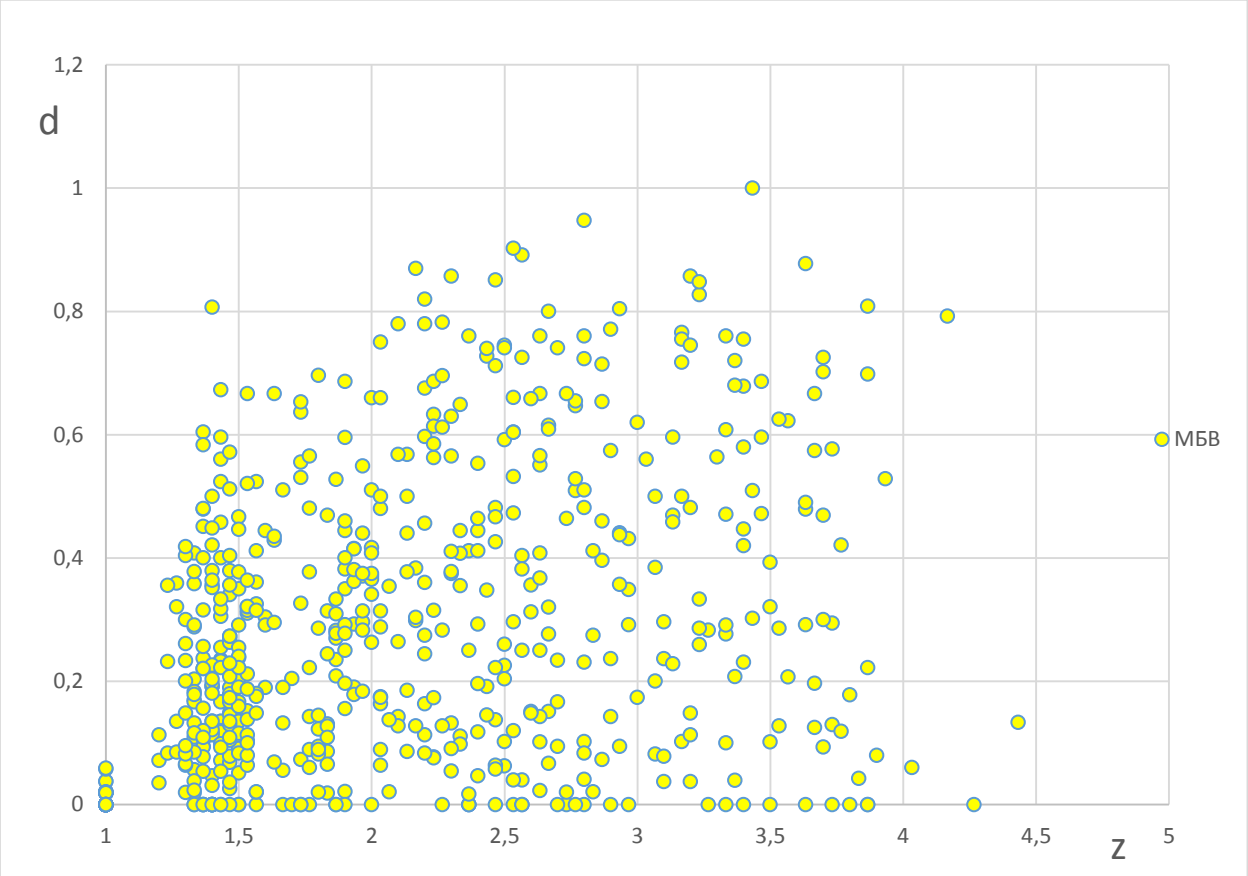

Рис. 3.8. Зависимость результата расстановки методом МБВ от сложности проекта

На рисунке 3.8 видно, что метод МБВ особенно хорошо работает при O~1, что соответствует ситуации, когда сетевой график проекта представлен цепочкой, кроме того, не видно явной связи эффективности с числом связей, о чем свидетельствуют наилучшие результаты, проявляющиеся на всем промежутке испытаний.

Таким образом, было установлено, что в диапазоне $1.25 < z < 1.9$ каждый метод может давать наилучший результат. Поэтому следующий этап будет проводиться в данном диапазоне.

Этап 2

Для проведения второго этапа испытания было сгенерировано 500 проектов с числом работ 30, числом контрольных точек $m = 3$, средней длительностью 7 д., разбросом длительностей 1 д. и со случайной структурой с числом связей меньше, чем 4. Далее из наблюдений были исключены все проекты, где сложность не входит в диапазон $1.25 < z < 1.9$. После чего в каждом проекте были расставлены КТ каждым из методов, и полученные значения критерия были сравнены между собой. Метод, давший наилучшее значение, отмечался победителем и заносился в соответствующий список. Если победителей было несколько, то данный результат не засчитывался. Далее было посчитано число отметок для каждого метода и нормировано. На основе данных значений был построен следующий график, представленный на рисунке 3.9.

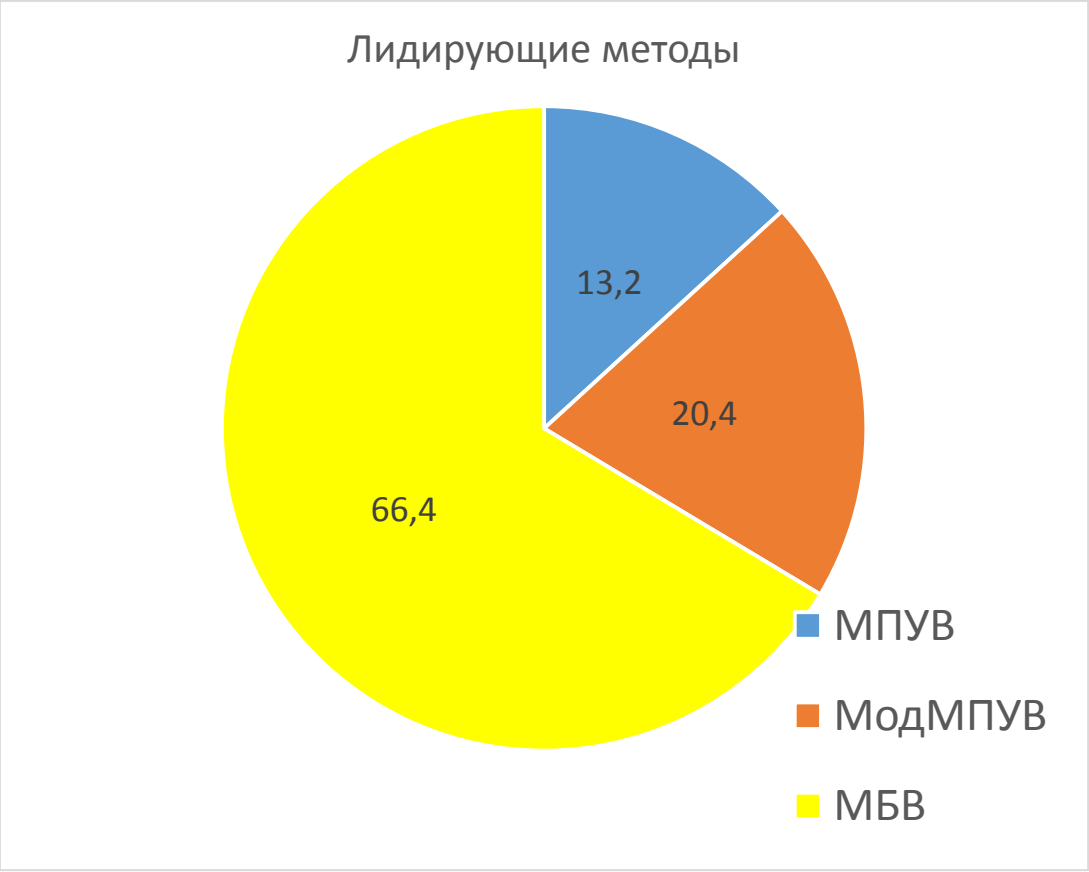

Рис. 3.9. Относительное число побед методов на наилучшем интервале

На основании полученных результатов, представленных на рисунке 3.9, можно сделать вывод, что метод МБВ давал наиболее часто наилучшие

результаты, однако существуют ситуации, когда данный метод дает результат хуже, чем МПУВ и МодМПУВ; кроме того, видно, что каждый метод имеет возможность дать наилучший результат, в связи с чем каждый из данных методов войдет в «комплекс» и будет исследоваться в дальнейшем.

Этап 3

Теперь можно оценить эффективность работы «комплекса». Для этого было сгенерировано 500 работ для двух ситуаций. Первая для числа работ 30 и 3 КТ на случайном значении сложности, вторая для числа работ 40 и 4 КТ на случайном значении сложности z. Полученные значения отклонений группировались по 10% начиная с 5% и выводились на график с частотой попадания в группы.

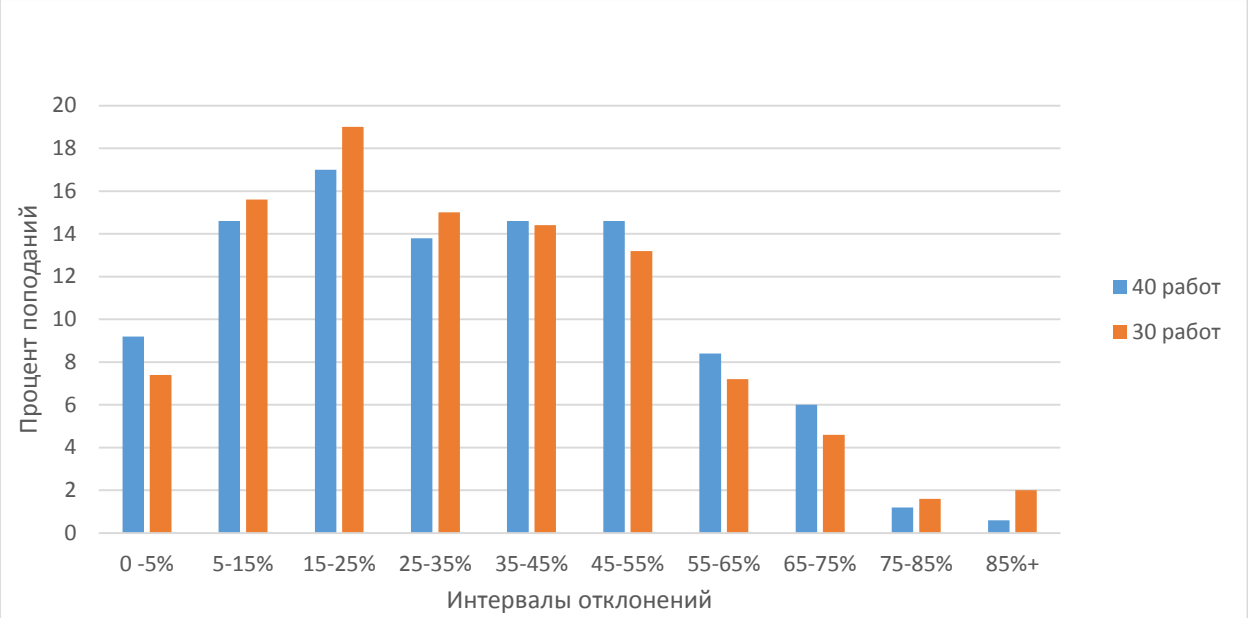

Рис. 3.10. Процент попаданий в интервал отклонения от оптимума

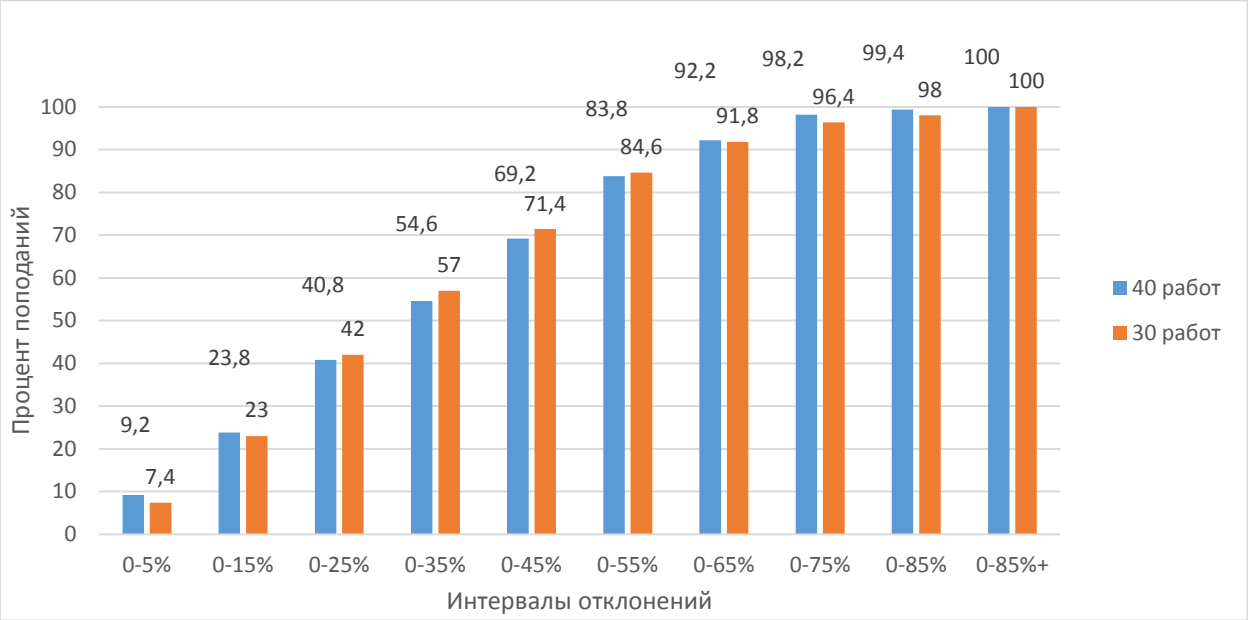

Рис. 3.11. Гистограмма отклонений от оптимума для «комплекса»

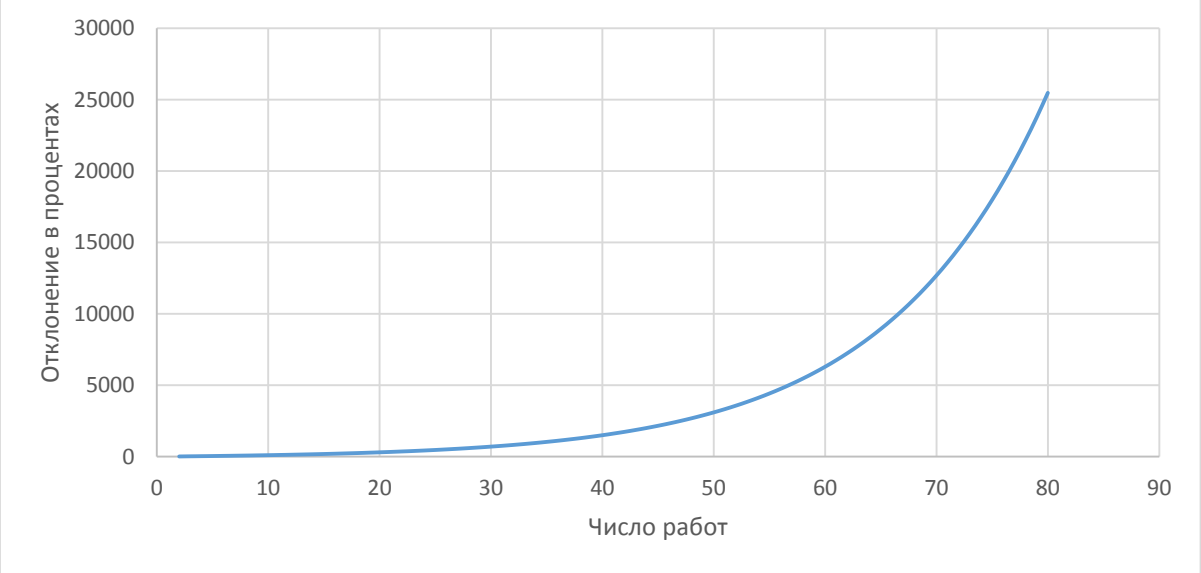

Рис. 3.12. Рост максимально возможного относительного отклонения от оптимума при росте числа работ

На рисунках 3.10. и 3.11 видно, что размер групп практически не меняется с увеличением числа работ в проекте, что позволяет сделать вывод о том, что при применении данного «комплекса» к реальным проектам, размещение контрольных точек будет находиться в таких же пределах. Кроме того, данные рисунки показывают, что 90% значений отклоняется от оптимума не более чем на 65%. На рисунке 3.12 особенно хорошо видно, что

при росте числа работ в проекте максимально возможные отклонения растут экспоненциально, что только увеличивает важность использования комплекса, дающего относительно небольшие отклонения.

3.4 Выводы по главе 3

На основании вышеизложенного можно сделать следующее выводы.

1. Решено оценивать эффективность предложенных методов через сравнение результатов, полученных ими после расстановок контрольных точек, с оптимальными расстановками, которые можно найти для небольших проектов методом полного перебора.
2. Сформированы условия проведения эксперимента и получены требования к разработке программного обеспечения для проведения эксперимента.
3. Разработан программный комплекс для проведения эксперимента, удовлетворяющий все полученные требования.
4. Произведен эксперимент с помощью разработанного программного комплекса в три этапа.
5. На первом этапе был получен интервал максимальной эффективности каждого метода в зависимости от логической сложности проектов.
6. На втором этапе было определено, что предложенные методы следует применять в комплексе. Доминирующего метода нет.
7. На третьем этапе была оценена эффективность работы «комплекса». Установлено, что значение критерия, полученного на основании расстановки «комплексом» с вероятностью 90%, отклоняется от оптимального не более чем на 65% при том, что диапазон возможного отклонения очень велик.

8. Вероятность отклонения от оптимума при применении предложенных методов не изменяется с ростом числа работ в проекте при том, что диапазон возможного отклонения стремительно растёт с ростом числа работ проекта.

9. Впервые был разработан программный комплекс для выбора контрольных точек в реальных проектах. Экспериментально доказанная эффективность заложенных в него методов, разработанных лично автором в данном диссертационном исследовании, даёт основание утверждать, что его применение практически полезно, поскольку позволяет за счёт своевременного обнаружения ошибок в работах уменьшить риск срыва сроков проекта. Пользователями разработанного программного комплекса являются руководители проектов, осуществляющие управление проектами с применением MS – Project.

Заключение

В ходе проведенного диссертационного исследования впервые получены следующие новые научные результаты.

1. Предложена диагностическая модель проекта (ДМП). Она позволяет представить соответствие между результатами проверок и возможными ошибками в работах. Для построения ДМП используется сетевой график проекта.

2. Разработана математическая модель возникновения ошибок в работах проекта. Предложена и модель расчёта вероятностей ошибок для ООП. В этом частном, но исключительно важно случае, выбор варианта расстановки контрольных точек уместно проводить в предположении о наличии ошибки в одной работе проекта.

3. Предложен критерий для сравнения наборов контрольных точек в ООП для случая, когда потери от задержки проекта нелинейно зависят от времени восстановления правильности выполнения проекта.

4. Предложен метод определения количества и мест расположения контрольных точек, который можно применить к большинству типовых проектов. В нём учитывается как возможность наличия ошибок в нескольких работах, так и затраты времени на выполнение операций контроля. Не только метод, но и сама задача выбора контрольных точек при наличии кратных ошибок не имеют аналога в литературе по технической диагностике.

5. Предложены методы выбора контрольных точек в ООП, совместное применение которых с последующим выбором наилучшего по предложенному автором минимаксному критерию позволяет получить вариант, более близкий к оптимальному. Методы используют, предложенные автором: диагностическую модель проекта (ДМП) и модель для расчёта вероятностей ошибок в ООП.

6. Предложены методы выбора контрольных точек в мульти-сценарных ООП, состав работ которых может измениться в ходе реализации.

7. Разработан программный комплекс, реализующий предложенные методы для реальных проектов. Кроме того, данный комплекс позволяет проводить испытания методов и «комплекса» для оценки эффективности.

8. Проведен эксперимент, направленный на оценку эффективности каждого из предложенных методов и «комплекса». Результаты эксперимента показали целесообразность использования всех трех методов в составе «комплекса», кроме того, высокую эффективность самого комплекса, которая не изменяется с ростом числа работ в проекте.

Список литературы

1. A Guide to the Project Management Body of Knowledge: PMBOK Guide 5th edition. – 2013. – Pennsylvania, Project Management Institute, 589 p.

2. Aleskerov, F. The threshold aggregation / F. Aleskerov, V. Chistyakov, V. Kalyagin // ECONOMICS LETTERS. – 2010. – № 2. – T. 107. – Pp. 261–262.

3. Andrew, T. Pro C# and the Net 4.5 Framework. 6 ed. / T. Andrew. – Apress, 2012. – 1600 p.

4. Perkins, B. Beginning C# 7 Programming with Visual Studio 2017 / B. Perkins, J. Hammer, J. Reid. – Paperback, 2018. – 912 p.

5. Biafore, B. Microsoft Project 2010: The Missing Manual / B. Biafore. – O'reilly Media, inc., 2010. – 770 p.

6. Cioffi, D. F. Completing projects according to plans: an earned-value improvement index / D. Cioffi // J. Oper. Res. Soc. – 2006. – T. 57. – № 3. – Pp. 290–295.

7. Coombs, C. R. When planned IS/IT project benefits are not realized: a study of inhibitors and facilitators to benefits realization / C. R. Coombs // Int. J. Proj. Manag. – 2015. – T. 33. – № 2. – Pp. 363–379.

8. Davis, R. Business Process Modeling with ARIS. A Practical Guide / R. Davis. – 2004. – London, Springer. – 531 p.

9. De Marco, A. Cost and schedule monitoring of industrial building projects: case study / A. De Marco, D. Briccarello, C. Rafele // J. Constr. Eng. Manag. – 2009. – T. 135. – № 9. – Pp. 853–862.

10. Futrell, R. Quality Software Project Management. – 1st Edition / R. Futrell, D. Shafer, L. Shafer. – Prentice Hall, 2002. – 1125 p.

11. GAPPS (2006) A Framework for Perfomance Based Competency Standarts for Global Level 1 and 2 Project Managers. Sydney: Global Alliance for Project Perfomance Standarts. – 2006. – 47 p.

12. Herroelen, W. Robust and reactive project scheduling: a review and classification of procedures / W. Herroelen, R. Leus // Int. J. Prod. Res. – 2004a. – T. 42. – № 8. – Pp. 1599–1620.

13. Herroelen, W. The construction of stable project baseline schedules / W. Herroelen, R. Leus // Eur. J. Oper. Res. – 2004b. – T. 156. – № 3. – Pp. 550–565.

14. Iain, D. Craig. The Interpretation of object-oriented programming languages / D. Iain // Springer. – 2nd edition (November 9, 2001). – 290 p.

15. Jahangirian, M. Simulation in manufacturing and business: A review / M. Jahangirian and others // Eur. J. Oper. Res. – 2010. – T. 203. – № 1. – Pp. 1–13.

16. Jaynes, E. T. Probability Theory: The Logic of Science / E. T. Jaynes. – 2003. – Cambridge: Cambridge University Press.

17. Johnson, B. Professional Visual Studio 2017 / B. Johnson. – John Wiley & Sons, 2017. – 864 p.

18. Jonson, R. An information theory approach to diagnostic / R. Jonson // Proc. 6-th National Symposium on Reliability and Quality Control. – 1960. – Pp. 102–109.

19. Kelton, W. Simulation with ARENA (Sixth edition) / W. Kelton, R. Sadowski. – NY: McGraw Hill Education, 2015.

20. Leus, R. Stability and resource allocation in project planning / R. Leus, W. Herroelen // IIEE Trans. – T. 36. – № 7. – 2004. – Pp. 667–682.

21. Lyneis, J. M. Strategic management of complex projects: a case study using system dynamics / J. M. Lyneis, K. G. Cooper, S. A. Els // Syst. Dyn. Rev. – T. 17. – № 3. – 2001. – Pp. 237–260.

22. Maron, M. A. The choice of control points of projects taking into account possible change of structure of works / M. A. Maron // Business Informatics. – 2016. – 2 (36). – Pp. 57–61.

23. Maron, M. A. Diagnostics of Projects / M. A. Maron // European Research Studies Journal. – 2018. – Vol. 21. – № 1. – Pp. 18–30.

24. OGC. Managing Successful Projects with PRINCE2: The Stationery Office, 2009. – 342 p.

25. Pritsker, A. A. B. The precedence GERT user's manual / A. A. B. Pristker. – Lafayete, IN: Pritsker & Associates, 1974.

26. Pritsker, A. A. B. GERT: Graphical Evaluation and Review Technique. – Part 1. Fundamentals / A. A. B. Pritsker, W. W. Happ // Journal of Industrial Engineering. – 1966. – Vol. 17. – № 5. – Pp. 267–274.

27. Pritsker, A. A. B. GERT: Graphical Evaluation and Review Technique. – Part 2. Applications / A. A. B. Pritsker, G. E. Whitehouse // Journal of Industrial Engineering. – 1966. – Vol. 17. – № 5. – Pp. 293–301.

28. Sakka, O. Relationship between the interactive use of control systems and the project performance: The moderating effect of uncertainty and equivocality / O. Sakka, H. Barki, L. Côté // Int. J. Proj. Manag. – Т. 34. – № 3. – 2016. – Pp. 508–522.

29. Shannon, C. E. A Mathematical Theory of Communication / C. E. Shannon // Bell System Technical Journal. – 1948. – Vol. 27. – Pp. 379–423.

30. Swartz, S. M. Managerial perceptions of project stability / S. M. Swartz // Proj. Manag. J. – 2008. – Т. 39. – № 4. – Pp. 17–32.

31. Taylor, H. Risk management and problem resolution strategies for IT projects: prescription and practice / H. Taylor // Proj. Manag. Q. – Т. 37. – № 5. – 2006. – P. 49.

32. Thakurta, R. Impact of Scope Creep on Software Project Quality / R. Thakurta // Vilakshan XIMB J. Manag. – 2013. – Т. 10. – № 1.

33. Williams, T. The Effects of Design Changes and Delays on Project Costs / T. Williams and others // J. Oper. Res. Soc. – T. 46. – № 7. – 1995. – P. 809.

34. Абрамов В. М. Критерии эффективности функционирования автоблокировки с учётом её надёжности/ В. М. Абрамов, А.П. Разгонов, Б.А. Давлетьяров // Вестник ВНИИЖТа. 1977. № 1. С. 51-54.

35. Авдошин, С. М. Информационные технологии управления рисками программных проектов / С. М. Авдошин, Е. Ю. Песоцкая // Информационные технологии. – 2008. – № 11. – С. 13–18.

36. Авдошин, С. М. Организационные риски при внедрении корпоративных систем и приложений / С. М. Авдошин, Е. Ю. Песоцкая // Качество. Инновации. Образование. – 2015. – № 3. – С. 47–53.

37. Акопов, А. С. Имитационное моделирование / А. С. Акопов. – М.: Юрайт, 2014. – 389 с.

38. Алескеров, Ф. Т. Бинарные отношения, графы и коллективные решения. – 2-е изд. / Ф. Т. Алескеров, Э. Л. Хабина, Д. А. Шварц. – М.: Физматлит, 2012. – 342 с.

39. Ананьин, В. И. Устойчивость управления IT-проектами в условиях неопределенности / В. И. Ананьин // Управление проектами. – 2005. – 1–2 (1–2). – М.: ИД «Гребенников»-СОВНЕТ.

40. Аньшин, В. М. Управление проектами с учетом концепции устойчивого развития / В. М. Аньшин // Научные исследования. Российский журнал управления проектами. – 2013. – № 2. – С. 3–15.

41. Артемьев, Д. Стратегическое управление проектами: цели, этапы, инструменты / Д. Артемьев, Т. Гергерт, Т. Пономарёва // Проблемы теории и практики управления. – 2013. – № 3. – С. 106–115.

42. Бараш, Л. Ю. Генерация случайных чисел и параллельных потоков случайных чисел для расчетов Монте-Карло / Л. Ю. Бараш,

Л. Н. Щур // Моделирование и анализ информационных систем. – 2012. – Т. 19. – С. 145–162.

43. Баркалов, С. Математические основы управления проектами / С. Баркалов, В. И. Воропаев, Г. И. Секлетова. – Москва: Высшая школа, 2005. – 423 с.

44. Башмачникова, Е. В. Элементы формирования теории управления проектами / Е. В. Башмачникова, Л. А. Абрамова // Вестник Поволжского государственного университета. Серия Экономика. – 2013. – № 2. – С. 125–133.

45. Беллман, Р. Динамическое программирование / Р. Беллман. – М.: Издательство иностранной литературы, 1960. – 400 с.

46. Богданов, В. Управление проектами. Корпоративная система – шаг за шагом / В. Богданов. – М: Манн, Иванов и Фербер, 2016. – 240 с.

47. Босс, В. Лекции по математике. Вероятность. Информация. Статистика / В. Босс. – М.: URSS: Ленанд, 2015. – 224 с.

48. Брейдо, А. И. Организация обслуживания железнодорожных устройств автоматики и связи / А. И. Брейдо, В. А. Овсяников. – М.: Транспорт, 1983. – 208 с.

49. Брускин, С. Н. Эффективные методы построения алгоритмов поиска неисправностей в информационных системах / С. Н. Брускин, А. А. Дружаев, А. И. Марон, М. А. Марон // Интеллектуальные системы в производстве. – 2017. – № 2. – С. 88–93

50. Вагнер, Г. Основы исследования операций. – Том 2 / Г. Вагнер. – М.: Мир, 1973. – 488 с.

51. Васильев, А. О. C#. Объектно-ориентированное программирование: Учебный курс / А. О. Васильев. – СПБ.: ПИТЕР, 2012. – 320 с.

52. Вентцель, Е. С. Исследование операций / Е. С. Вентцель. – М.: Советское радио, 1972. – С. 200–206.

53. Володин, С. В. Функционально-структурные особенности стратегического управления проектами / С. В. Володин // Российское предпринимательство. – 2013. – № 4. – С. 59–68.

54. Воробьева, О. А. Кризисное управление в проектной деятельности / О. А. Воробьева // Менеджмент: теория и практика. – 2012. – № 1/2. – С. 110–114.

55. Воропаев, В. И. Системный подход к управлению проектами и программами / В. И. Воропаев, Г. И. Секлетова // Управление проектами и программами. – № 3. – 2005. – С. 20–29.

56. Воскобойников, Ю. Основы вычислений и программирования в пакете MathCAD PRIME / Ю. Воскобойников, А. Задорожный. – Санкт-Петербург: Лань, 2016.

57. Гнеденко, Б. Математические методы в теории надёжности / Б. Гнеденко, Ю. Беляев, А. Соловьёв. – М.: URSS, 2013.

58. ГОСТ Р ИСО 21500-2014 Руководство по проектному менеджменту. – М.: Стандартинформ, 2015. – 52 с.

59. Грачёв, Н. Н. Способ контроля качества сборки блоков радиоэлектронных средств / Н. Н. Грачёв // Инновационные информационные технологии. – 2013. – № 2. – Т. 2. – С. 160–166.

60. Гриненко, А. В. Автоматизированная обучающая система для дистанций сигнализации и связи / А. В. Гриненко, В. В. Нестеров, В. Л. Лабецкий // Автоматика, связь, информатика. – 2001. – № 11. – С. 22–25.

61. Громов, А. И. Управление бизнес-процессами. Современные методы / А. И. Громов, А. Фляйшман, В. Шмидт. – М: Юрайт, 2016. – 368 с.

62. Гуреева, Е. В. Управление проектами. Стратегическое планирование / Е. В. Гуреева, М. В. Недовесов // Системы управления и информационные технологии. – 2012. – № 2. – С. 95–98.

63. Гурин, В. Диагностика автоматизированного производства / В. Гурин и др. – М.: Машиностроение, 2011. – 600 с.

64. Демидович, Б. П. Основы вычислительной математики. – 5-е изд. / Б. П. Демидович, И. А. Марон. – СПб.: Лань, 2006. – 672 с.

65. Джаафари, А. Диагностика проекта как средство ускорения его реализации / А. Джаафари, Н. Джабари // Управление проектами и программами. – № 2. – 2008. – С. 128–139.

66. Духин, А. А. Теория информации / А. А. Духин. – М.: Гелиос АРВ, 2007. – 248 с.

67. Дмитренко, И. Е. Измерения и диагностирование в системах железнодорожной автоматики, телемеханики и связи / И. Е. Дмитренко, В. В. Сапожников, Д. В. Дьяков. – М.: Транспорт, 1994. – 283 с.

68. Ефанов, Д. В. Обеспечение безопасности движения за счет технического диагностирования и мониторинга устройств железнодорожной автоматики и телемеханики / Д. В. Ефанов, П. А. Плеханов // Транспорт Урала. – 2011. – № 3. – С. 44–48.

69. Ефанов, Д. В. Построение оптимальных алгоритмов поиска неисправностей в технических объектах / Д. В. Ефанов. – СПб.: ФГБОУ ВПО ПГУПС, 2014. – 49 с.

70. Зажигаев, Л. С. Методы планирования и обработки результатов физического эксперимента / Л. С. Зажигаев, А. А. Кишьян, Ю. И. Романников. – М.: Атомиздат, 1978. – 232 с.

71. Златопольский, Д. М. Программирование: типовые задачи, алгоритмы методы. – 2-е изд. / Д. М. Златопольский. – М.: БИНОМ. Лаборатория знаний, 2012. – 223 с.

72. Зуйков, К. А. Устойчивость проекта. Подход, основанный на системной динамике / Е. А. Зуйков // Управление проектами и программами. – 2012. – № 3 (31). – С. 178–187.

73. Ильин, В. В. Руководство качеством проектов. Практический опыт / В. В. Ильин. – М.: Вершина, 2006. – 176 с.

74. Ильина, О. Управление проектами: ориентация на устойчивое развитие / О. Ильина // Проблемы теории и практики управления. – 2012. – № 1. – С. 106–112.

75. Ирзаев, Г. Х. Исследование и моделирование информационных потоков конструкторско-технологических изменений на этапах освоения и серийного производства изделий / Г. Х. Ирзаев // Организатор производства. – 2012. – № 1. – С. 131–135.

76. Исаев, Д. В. Моделирование реализации проектов внедрения аналитических информационных систем / Д. В. Исаев // Аудит и финансовый анализ. – 2014. – № 6. – С. 416–422.

77. ИСО/ТО 10006:1997 (Е). Менеджмент качества. Руководство качеством при управлении проектами. – М., 1997. – 39 с.

78. Казак, А. Ю. Современные методы оценки проектных рисков: традиции и инновации / А. Ю. Казак, Ю. Э. Слепухина // Вестник УРФУ. Серия Экономика и управление. – 2013. – № 2. – С. 13–26.

79. Карибский, В. В. Основы технической диагностики: (Модели объектов, методы и алгоритмы диагноза) / В. В. Карибский, П. П. Пархоменко, Е. С. Согомонян, В. Ф. Халчев. – М.: Энергия, 1976. – 464 с.

80. Каштанов, В. А. Теория массового обслуживания / В. А. Каштанов, Г. И. Ивченко, И. Н. Коваленко. – М.: Либроком, 2012.

81. Каштанов, В. А. Теория надежности сложных систем / В. А. Каштанов, А. И. Медведев; рук. В. А. Каштанов. – М.: Физматлит, 2010.

82. Кирпичников, А. Методы прикладной теории массового обслуживания / А. Кирпичников. – М.: URSS, 2018.

83. Клюев, В. В. Технические средства диагностирования: Справочник / В. В. Клюев, П. П. Пархоменко, В. Е. Абрамчук и др.; под общ. ред. В. В. Клюева. – М.: Машиностроение, 1989. – 672 с.

84. Кнут, Д. Искусство программирования. – Т. 3. Сортировка и поиск. – 2-е изд. / Д. Кнут. – М.: ООО «И. Д. Вильямс», 2017. – 832 с.

85. Коньшунова, А. Ю. К вопросу о классификации проектов в проектном управлении / А. Ю. Коньшунова // Экономика и современный менеджмент: теория и практика. – 2013. – № 32. – С. 171–178.

86. Копылова, О. В. Анализ рисков в процессе управления проектами / О. В. Копылова // Российское предпринимательство. – 2013. – № 11. – С. 44–48.

87. Кравченко, Т. К. Управление требованиями при реализации ИТ-проектов / Т. К. Кравченко // Бизнес-информатика. – 2013. – № 3. – С. 63–71.

88. Крук, Е. А. Методы программирования и прикладные алгоритмы: учеб. пособие: в 3 ч. / Е. А. Крук, А. Овчинников. – СПб.: ГУАП, 2014.

89. Кудряшов, Б. Д. Теория информации / Б. Д. Кудряшов. – Санкт-Петербург: СПбГУ ИТМО, 2010. – 188 с.

90. Кузьмин, Е. А. Идентификация рисков в управлении проектами методом анализа балансов факторов и отклонений / Е. А. Кузьмин // Управление финансовыми рисками. – 2012. – № 3. – С. 200–214.

91. Лесных, В. В. Об оценке значимости выполнения работ проектов / В. В. Лесных, Ю. В. Литвин // Аудит и финансовый анализ. – 2013. – № 4. – С. 254–260.

92. Липаев, В. В. Программная инженерия. Методологические основы: Учебник / В. В. Липаев. – М.: Гос. ун-т – Высшая школа экономики. – ТЕИС, 2006. – 609 с.

93. Лисенков, В. М. Статистическая теория безопасности движения поездов / В. М. Лисенков. – М: ВИНИТИ РАН, 1999. – 332 с.

94. Магомаева, Л. Анализ и классификация всех методов управления рисками при управлении инновационными проектами / Л. Магомаева // Предпринимательство. – 2013. – № 5. – С. 130–141.

95. Макконнелл, С. Совершенный код: практическое руководство по разработке программного обеспечения / С. Макконнелл; пер. с англ., русская редакция. – 2010. – 867 с.

96. Маклаков, С. В. Создание информационных систем с AllFusion Modeling Suite / С. В. Макланов. – М.: ДИАЛОГ-МИФИ, 2005. – 432 с.

97. Малкин, В. С. Техническая диагностика / В. С. Малкин. – СПб: Лань, 2008. – 272 с.

98. Марон, А. И. Информационный подход к организации контроля проектов / А. И. Марон, М. А. Марон // Бизнес-информатика. – 2012. – № 4. – С. 54–60.

99. Марон, А. И. Оптимизация контроля в программных проектах разработки больших систем / А. И. Марон, М. А. Марон // Материалы XXXVIII Международной конференции «Информационные технологии в науке, социологии и бизнесе. IT+S&E' 10». – Крым: Ялта-Гурзуф, 2010. – 20–30 мая. – 46–48 с.

100. Марон, В. И. Статистические модели на основе информационного подхода Джейнса / В. И. Марон. – М.: МАКС-ПРЕСС, 2011. – 156 с.

101. Марон, М. А. Дискретно-событийная имитационная модель реализации проекта / М. А. Марон // Глава в книге: Труды ежегодной международной научно-практической конференции «Новое в науке и образовании». – М.: МАКС-ПРЕСС, 2016. – С. 272–278.

102. Марон, М. А. Определение размера репрезентативной выборки и ее реальное применение / М. А. Марон // В кн.: Статистические методы анализа экономики и общества: Тезисы докладов Межвузовской

студенческой научно-практической конференции (13–14 мая 2010 г.). – М.: Высшая школа экономики, 2010. – С. 58–59.

103. Марон, М. А. Снижение рисков проектов путем использования методов технической диагностики. Выпускная квалификационная работа магистра / М. А. Марон. – М.: НИУ ВШЭ, 2014.

104. Марон, М. А. Использование энтропийного подхода в диагностике программных проектов. Выпускная квалификационная работа бакалавра / М. А. Марон. – М.: НИУ ВШЭ, 2012.

105. Матяш, И. В. Диагностика проекта: анализ динамики уровня затрат / И. В. Матяш // Известия Алтайского государственного университета. – 2-2 (78). – 2013. – С. 269–273.

106. Махтева, И. П. Особенности методов и инструментов снижения рисков при управлении инновационными проектами в холдинге / И. П.Махтева // Экономика и предпринимательство. – 2013. – № 1. – С. 467–472.

107. Михеев, Р. Н. VBA и программирование в MS Office для пользователей / Р. Н. Михеев. – СПБ.: БХВ-Петербург, 2006. – 384 с.

108. Пархоменко, П. П. Основы технической диагностики: (Оптимизация алгоритмов диагностирования, аппаратурные средства) / П. П. Пархоменко, Е. С. Согомонян. – М.: Энергия, 1981. – 320 с.

109. Патрушева, А. А. Технология использования системы управления проектами по разработке программного обеспечения / А. А. Патрушева // Вестник Иркутского государственного технического университета. – 2013. – № 5. – С. 185–189.

110. Песелис, А. Управление рисками на предприятии в проектах разработки программного обеспечения / А. Песелис // Предпринимательство. – 2012. – № 6. – С. 136–144.

111. Петракова, В. А. Модели и алгоритмы решений в управлении проектом / В. А. Петракова, А. С. Сомова // Известия Южного федерального университета. Технические науки. – 2012. – № 5. – С. 122–127.

112. Петросянц, К. О. Моделирование сбоев в КНИ/КНС КМОП-схемах с использованием универсальной SPICE-модели / К. О. Петросянц, Л. М. Самбурский, И. А. Харитонов // В кн.: XVI Всероссийская научно-техническая конференция «Электроника, микро- и наноэлектроника»: 3–7 июля 2017 года, г. Суздаль, Россия. – М.: НИИСИ РАН, 2017. – С. 53–54.

113. Рихтер, Д. CLR via C#. Программирование на платформе Microsoft.NET Framework 4.5 на языке C# / Д. Рихтер. – СПб.: ПИТЕР, 2017. – 896 с.

114. Рудакова, А. А. Сравнительный анализ традиционной методологии проектного управления и гибкого управления инновационными проектами / А. А. Рудакова // Экономика и предпринимательство. – 2012. – № 6. – С. 292–294.

115. Сапожников, Вл. В. Основы технической диагностики / Вл. В. Сапожников, В. В. Сапожников. – М.: Маршрут, 2004.

116. Соколов, М. Ю. Управление рисками в проектах государственно-частного партнерства / М. Ю. Соколов, С. В. Маслова // Вестник Санкт-Петербургского университета. Серия 8. Менеджмент. – 2013. – № 4. – С. 100–124.

117. Тебеньков, А. Н. Оценка рисков в управлении проектами / А. Н. Тебеньков // Менеджмент: теория и практика. – 2012. – № 3/4. – С. 107–114.

118. Терехова, А. Е. Проблемы управления большими и сложными проектами / А. Е. Терехова, Н. Ю. Верба // Вестник Университета (Государственный университет управления). – 2013. – № 2. – С. 161–165.

119. Тихоненко, К. П. Инновации и управление проектами: исследование связей / К. П. Тихоненко, Н. И. Меркушова // Проблемы современной экономики. – 2013. – № 16. – С. 59–63.

120. Толстяков, В. С. Обнаружение и исправление ошибок в дискретных устройствах / В. С. Толстяков. – М.: Советское радио, 1972. – 288 с.

121. Томорадзе, И. Управление проектами как стадия процессного управления / И. Томорадзе, А. Дмитрик // Проблемы теории и практики управления. – 2013. – № 2. – С. 93–100.

122. Трофимов, В. В. Управление проектами с PRIMAVERA / В. В. Трофимов и др. – СПб.: Издательство СПбГУЭФ, 2006. – 216 с.

123. Трухановский, О. М. Анализ исторического развития офисов управления проектами в современных инновационных компаниях / О. М. Трухановский // Экономика и предпринимательство. – 2012. – № 2. – С. 120–122.

124. Фатрелл, Р. Управление программными проектами. Достижение оптимального качества при минимуме затрат / Р. Фатрелл, Д. Шафер, Л. Шафер. – Вильямс, 2003. – 1125 с.

125. Филипс, Дж. Менеджмент IT-проектов. На пути от старта до финиша / Дж. Филипс. – М., 2005. – 375 с.

126. Хаммер, М. Реинжиниринг корпорации: Манифест революции в бизнесе / М. Хаммер, Дж. Чампи. – СПб.: Издательство СПбГУ, 1999.

127. Хемди А. Таха Х. Исследование операций / Хемди А. Таха Х. – М.: Вильямс, 2016. – 912 с.

128. Царьков, И. Н. Классификация математических моделей управления проектами / И. Н. Царьков // Научные исследования и разработки. Российский журнал управления проектами. – 2015. – Т. 4. – № 1. – С. 13–19.

129. Ципес, Г. Л. Проекты и управление проектами в современных компаниях / Г. Л. Ципес, А. С. Товб. – М.: Олимп-Бизнес, 2009. – 480 с.

130. Чатфилд, К. Microsoft Project 2013. Шаг за шагом / К. Чатфилд, Д. Джонсон. – М.: ЭКОМ Паблишерз, 2013. – 672 с.

131. Черемных, О. С. Стратегический корпоративный реинжиниринг: процессно-стоимостной подход к управлению бизнесом / О. С. Черемных, С. В. Черемных. – М. Финансы и статистика, 2005.

132. Чернавский, Д. С. Синергетика и информация. Динамическая теория информации / Д. С. Чернавский. – М.: Либроком, 2016. – 302 с.

133. Шеннон, К. Работы по теории информации и кибернетике / К. Шеннон. – М.: Издательство иностранной литературы, 1963. – 830 с.

134. Шишкин, Е. В. Исследование операций / Е. В. Шишкин. – М.: Проспект, 2008. – 208 с.

135. Шишмарев, В. Диагностика и надежность автоматизированных систем / В. Шишмарев. – М.: Academia, 2013. – 352 с.

136. Якимович, Б. А. Теоретические основы конструктивно-технологической сложности изделий и структур-стратегий производственных систем машиностроения / Б. А. Якимович, А. И. Коршунов, А. П. Кузнецов. – Ижевск: Издательство ИжГТУ, 2007. – 280 с.