Анатолий Косолапов

Концептуальное проектирование компьютерных систем реального времени

CoDeCS - задачи, модели, методы, алгоритмы, программы

LAP LAMBERT Academic Publishing

Анатолий Косолапов

Концептуальное проектирование компьютерных систем реального времени

Анатолий Косолапов

Концептуальное проектирование компьютерных систем реального времени

CoDeCS - задачи, модели, методы, алгоритмы, программы

LAP LAMBERT Academic Publishing RU

Imprint

Cover image: www.ingimage.com

Publisher:
LAP LAMBERT Academic Publishing
is a trademark of
International Book Market Service Ltd., member of OmniScriptum Publishing Group
17 Meldrum Street, Beau Bassin 71504, Mauritius

Printed at: see last page
ISBN: 978-620-0-08208-4

Содержание

Перечень условных обозначений 4

1. Анализ современных направлений, основных пробдем и резервов развития АСУ 7

1.1 Развитие парадигмы компьютеризации 7

1.2 Архитектура систем автоматизации. 9

1.3 Общие тенденции и этапы усовершенствования архитектуры автоматизованих систем 12

1.4 Развитие архитектуры станционных систем автоматизации в Украине 19

1.5 АСУ ГП УЗ-Е как основа интеграции информационно-управляющих систем железных дорог Украины 21

1.5.1 Четырех уровневая структура построения системы 21

1.5.2 Вычислительные и коммуникационные ресурсы АСУ ГП УЗ-Е 24

1.5.3 Особенности построения и функционирования системы 25

1.5.4 "Предельные" проектные оценки характеристик системы 25

1.5.5 Оценка эксплуатационных характеристик АСУ ГП УЗ-Е 28

2. Развитие комплексного подхода к системному анализу, проектированию и совершенствованию систем автоматизации 31

2.1 Особенности проектирования и эксплуатации сложно структурированных систем управления реального времени 31

2.2 Современные тенденции развития процессов информатизации и их влияние на проектирование информационных систем 37

2.3 Принципы и общие требования к методологии проектирования систем реального масштаба времени 40

2.4 Методики концептуального проектирования как основа создания сложных ресурсосберегающих АСУ 42

2.5 Формирование научно-методического комплекса системного интегратора (КСИ - Ξ) CoDeCS **(Co**nceptual **De**sign of **C**omplex Real-time Management **S**ystem) 47

3. Методы построения и исследования информационно-функциональных и технических структур АСУ 55

3.1 Принципы построения М-моделей распределенных информационно-управляющих систем 55

3.2 Задача структурной оптимизации М-модели и общий алгоритм ее решения 59

3.3 Методы поиска рациональных вариантов информационных структур АСУ на основе генетических алгоритмов 62

3.3.1 Постановка задачи и метод рационального распределения информационных потоков в информационно-управляющих систмах 62

3.3.2 Метод поиска рациональных "звездообразных" структур информационных систем с использованием кодирования Прюфера 74

3.4 Ресурсосберегающие методы выбора технических структур децентрализованных информационно-управляющих систем 84

3.4.1 Метод исследования условий целесообразности децентрализации функций управления в системах горочной автоматики 84

3.4.2 Метод выбора технических структур цифровых управляющих систем (ЦУС) по критерию эффективного использования вычислительных ресурсов 94

3.4.3 Подход к оценке затрат на построение распределенных ЦУС 106

4. Развитие моделей и методов расчёта информационно-временных характеристик АСУ реального времни 111

4.1 φ - транзакция как основная модель для оценки информационно-временных характеристик сервис-ориентированных систем 111

4.1.1 Определение понятия φ - транзакция 111

4.1.2 Метод оценки характеристик ЦУС реального времени в процессе проектировния 113
4.2 Методы обработки φ - транзакций и расчет их характеристик 121
4.2.1 Метод оценки математического ожидания времени выполнения φ - транзакций 121
4.2.2 Автоматизация построения и расчета характеристик φ - транзакций 126
4.3 Методика, аналитические модели и инструментальные средства выбора системы приоритетов обработки φ -транзакций 128

5. Усовершенствование методов анализа и расчета надежности систем с нечеткими параметрами 137
5.1 Метод анализа надежности нечетких систем с использованием теории размытых мнжеств 137
5.2 Метод использования различных видов размытых множеств для оценки надежности нечетких систем 147
5.3 Примеры решения задач оценки надёжности нечётких систем без резервирования и с многократным резервом 153

СПИСОК ЛИТЕРАТУРЫ 161

Перечень условных обозначений

АСУ	автоматизированная система управления
АСУ ГП УЗ-Е	единственная АСУ грузовыми перевозками "Укрзалізниці"
АСУ МД	АСУ маршрутами движения на сортировочной станции (рос. АСУ МД)
АСУ РСГ	АСУ расформирования составов на горке (рос. АСУ РСГ)
АСУ СС	АСУ сортировочной станции (рос. АСУ СС)
АСОТУСС	автоматизированная система организационно-технологического управляння сортировочной станции
АС УГП	аналитические серверы управления грузовыми перевозками
АРС	система автоматизированного управления скоростью отцепов на горке
АП	абсолютные приоритеты
БД	база даных
БП	безприоритетная дисциплина обслуживания заявок
КРИХС	контроль расцепа и измерение ходовых свойств
ОП	относительные приоритеты
ДСРМВ	дискретные СРМВ
ЭВМ	электронная вычислительная машина
СП	смешанные приоритеты
ИАРС	автоматическое регулирование скорости отцепов в интервальных тормозных позициях
ИУК СС	информационно-управляющий комплекс сортировочной станции (рос. ИУК СС)
ИУВС	информационно-управляющая вычислительная система
ИПС СС	информационно-планирующая система сортировочной станции
УВК	управляющий вычислительный комплекс (рус. УВК)

УВМ	управляющая вычислительная машина (рус. УВМ)
КПСА	комплекс программно-технических средств автоматизации
КСИ	комплекс системного интегратора
КТС	комплекс технических средств
ЛО	лингвистическое обеспечение
МО	математическое обеспечение
МКЭВМ	микро-ЭВМ
МКК	микроконтроллер
МККпс	микроконтроллерная подсистема
МКП	микропроцессор
НО	напольное (наземное) оборудования
НА	низовая автоматика
ОО	организационное обеспечение
ОС	операционная система
АРСП	автоматическое регулирование скорости отцепов в прицельных тормозных позициях
ПО	программное обеспечение
УСО	устройства связи с объектом (рус. УСО)
РИУВК	распределенная ИУВК
СМО	система массового обслуживания
СРМВ	система реального масштаба времени
ТЗ	техническое задание
ТОА	технологический объект автоматизации
ТОУ	технологический объект управления
УИМ	условная информационная мощность АСК
ФПБ	функциональный программный блок
ФВД	функционально-временные диаграммы
ЦУС	цифровая управляющая система

Capacity reserves	производительность, пропускная способность, емкость
HC-FS	Hard Client - File Server (Толстый клиент - Файловый сервер)
HC-DBS	Hard Client - Data Base Server (Толстый клиент - сервер баз данных)
p2p-архітектура	peer to peer архитектура; одноранговая, децентрализованная или пиринговая сеть
rsДСРМЧ	реактивная ДСРМЧ
SC-AS-DBS	Slim Client - Application Server - Data Base Server (Тонкий клиент - Сервер приложений - Сервер баз данных)
SWC-WS-DBS	Slim Web Client - Web Server - Data Base Server (Тонкий веб-клиент - Веб-сервер - Сервер баз данных)
SWC-WS-AS-DBS	Slim Web Client - Web Server - Application Server - Data Base Server (Тонкий веб-клиент - Веб-сервер - Сервер приложений - Сервер баз данных)
“The bottom line”	верхние расходы, полные проектные стоимости
Timeliness	своевременность

1. Анализ современных направлений, основных пробдем и резервов развития АСУ

1.1 Развитие парадигмы компьютеризации

Развитие средств вычислительной техники и информационных технологий в последнее время привело к изменению парадигмы в понимании сущности и основных требований к системам. Напомним, что парадигма (с греческого «пример, модель, образец») - это совокупность фундаментальных научных установок, представлений и терминов, принимается и разделяется сообществом и объединяет большинство его членов. Она обеспечивает преемственность развития науки и научного творчества. Фундаментальное представление об информационных системах можно свести к иерархии основных требований к ним (см. рис. 1.1).

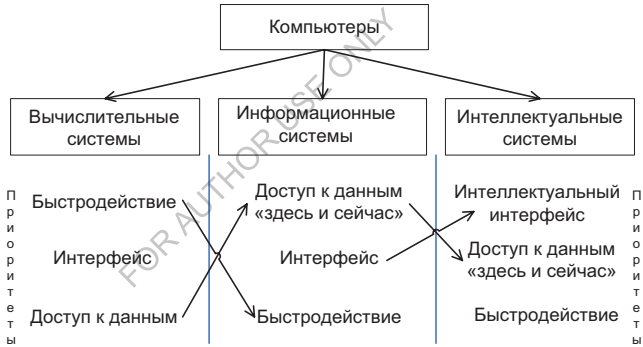


Рис.1.1 Развитие парадигмы компьютерных систем

На ранних этапах развития ЭВМ и систем основным требованием пользователей было обеспечение максимальной скорости ВЫЧИСЛЕНИЙ или быстродействия при решении задач измерения, контроля и управления. Поэтому все вели разговоры про ВЫЧИСЛИТЕЛЬНЫЕ системы и системы автоматизации АСУП, АСУТП, ИУК и т.п. Вопрос интерфейса и доступа к данным были на втором плане.

В начале нового века, которые стали называть информационным, сеть ИНТЕРНЕТ стала не только транспортным средством, но и большим информационным ресурсом, появились WEB-системы и произошла смена парадигмы и иерархии требований: заговорили про ИНФОРМАЦИОННЫЕ

СИСТЕМЫ, которые объединяют все типы компьютерных систем независимо от решаемых задач. В таких системах основным требованием является доступ к информации «здесь и сейчас». Сетевые технологии предоставили возможность работы в реальном масштабе времени для территориально-распределенных систем.

Появление новой парадигмы наблюдается в наше время, когда пользователей не беспокоит быстродействие систем, доступ к данным, но проявляется растущая потребность в качестве ожидаемый данных, которые обладали бы релевантностью, достоверностью и помогали бы в процессе принятия решений. В данном случае речь идет про интеллектуальный интерфейс, идея которого была впервые предложена японцами в проекте стратегических вычислительных машин пятого поколения в 1981 году. Такая машина была построена на базе моделей и методов искусственного интеллекта, в качестве базового языка использовался Пролог. Новая система была в одном экземпляре изготовлена в 1991 году, но оказалась очень дорогой и невостребованной в начале 90-х. И только в наше время начали активно проводиться работы по созданию компьютерных систем с элементами искусственного интеллекта, или интеллектуальных систем. В новой парадигме на первый план выходят вопросы интеллектуализации интерфейса и решения прикладных задач с использованием моделей и методов искусственного интеллекта.

Перспектива решения многих проблем - в переходе к интеллектуальному железнодорожному транспорту, сочетающий взаимодействие «умного» локомотива и «умной» станции. Интеллектуальные железнодорожные системы получают все большее распространение в мировой практике, их разработкой занимаются ведущие мировые фирмы. Создание и внедрение таких систем поддерживаются международными транспортными организациями. Интеллектуальные технические средства позволяют облегчить работу, обеспечить логический контроль за его действиями в штатных и нештатных ситуациях. С их помощью есть возможность проводить расширенную и оперативную диагностику работы оборудования и принимать решения по обеспечению надежности, безопасности и жизнеспособности процесса перевозок [17].

1.2 Архитектура систем автоматизации

Сначала необходимо определить понятие «архитектура» как базовое понятие любой системы.

Системная архитектура, по стандарту ANSI / IEEE 1471-2000is, - это «фундаментальная организация системы, реализованная в ее компонентах, связи этих компонентов друг с другом и внешней средой и принципах, определяющих структуру и развитие системы». Фактически в этом определении речь идет о совокупности различных структур.

Это подтверждает и анализ материалов различных источников, выполненный в [45]: термин «архитектура системы» часто является синонимом термина «структура системы». При использовании термина «архитектура системы» на первый план выдвигается сложный, многоаспектный характер структуры системы. Вот некоторые наиболее известные варианты определения термина (понятия) «архитектура системы» (см. таблица 1.1).

Таблица 1.1

Определение понятия архитектура информационной системы

Источник	Определение
IEEE Recommended Practice for Architectural Description, Draft 3.0 of IEEE P1471, May 1998	Архитектура - высокоуровневая концепция системы, которая учитывает окружения
ISO-15704, Industrial automation systems – Requirements for enterprise-reference architectures and methodologies. August 20, 1999	Архитектура системы - описание (модель) основного расположения и взаимосвязей частей системы (физического или концептуального объекта или существа)
ANSI/IEEE Std 1471-2000, Recommended Practice for Architectural Description of Software-Intensive Systems	Архитектура - фундаментальная организация системы, помещенная в своих компонентах, в их взаимоотношениях, в окружении, а также **принципы, определяющие проектирование, создание и развитие системы**

Источник	Определение
Международный центр научной и технической информации »- http://www.icsti.su	Архитектура - концепция, которая определяет структуру и взаимосвязь компонентов сложного объекта
Словник http://www.glossary.ru	Архитектура информационной системы - концепция, определяющая модель, структуру, функции и взаимосвязь компонентов информационной системы

Отечественные стандарты и нормативные документы [4] не используют термин «архитектура системы». Но в них определяются:

- Виды структур ИС - функциональная, техническая, организационная, программ-на, информационная;

- Основные структурные компоненты ИС - пользователи и комплекс средств автоматизации (КСА);

- Виды обеспечения ИС – комплекс технических средств (КТС), программное, информационное, организационное, методическое, математическое, лингвистическое, правовое и др .;

- Необходимость выделения структуры функциональных систем и подсистем ИС, описания состава и характеристик автоматизированных функций и задач ИС.

Фактически, любая архитектура ИС, независимо от парадигмы - это множество взаимосвязанных структур, которая описывается следующим выражением:

$$АИС = Цель \cap (КТС \cup МО \cup ПО \cup ИО \cup ЛО \cup ОО \cup МетрО \cup ДО) \quad (1.1)$$

где КТС или ТО (техническое обеспечение) - комплекс технических средств системы; ПО - программное обеспечение (общее и специальное ПО); МО - математическое обеспечение (совокупность математических моделей, методов и алгоритмив); ИО - информационное обеспечение (описание сигналов, принципов классификации и кодирования информации, описание массивов, форм, нормативно-справочной и других видов информации);

ЛО - лингвистическое обеспечение (совокупность языковых средств общения персонала с системой); ОО - организационное обеспечение (организационная структура, оперативный персонал, должностные инструкции персоналу); МетрЗ - метрологическое обеспечения (средства обеспечения

заданных достоверных характеристик измерительных функций системы); ДО – комплект документации на систему и все виды обеспечения; Цель - цель создания системы. Все эти виды обеспечение характеризуются набором взаимосвязанных статических и динамических структур, которые формируются в процессе проектирования системы и объединены общей концептуальной схемой для достижения целей создания при минимизации суммарных затрат [78, 140].

В зависимости от парадигмы доминирует та или иная часть в данном составе видов обеспечения. Для *вычислительных систем* - это ТС (быстродействие), ПО (организация вычислений), для *информационных систем* - ИО (базы данных), ПО (сетевое программное обеспечение), ТС (сетевые структуры), для интеллектуальних систем - МО (модели и методы представления и обработки знаний), ИО (базы знаний), ЛО (языка интеллектуального общения), ПО (языки логического и функционального программирования, оболочки экспертных сис-тем), ОО (пользователи системы), ТС (специальные устройства ввода-вывода).

Предложенную многозвеньевую формулу архитектуры ИС можно сравнять с поездом, в котором есть локомотив, который обеспечивает движение к поставленной цели - это КТС. Мы не можем «делать более тяжелыми» виды обеспечений в «поезде» без согласования с возможностями КТС. Например, использование сложных алгоритмов управления скоростью скатывания отцепов в тормозених позициях на основе моделирования физических процессов движения отцепов на горке ограничиваются быстродействием контроллеров, которые используются. Наращивание количества задач в действующих системах или увеличение количества клиентов, требуют обслуживания, опасно без учета нужных вычислительных ресурсов и пропускной способности каналов. Игнорирование таких оценок приводит к отказам систем при критических нагрузках или к снижению качества принимаемых решений. Поэтому выбор «локомотива» для поддержания соответствующего «состава» видов обеспечения является важной задачей, как при проектировании, так и при модернизации интеллектуальных систем на станциях.

1.3 Общие тенденции и этапы усовершенчтвования архитектури автоматизированных систем

Архитектуру информационной системы автоматизированного управления можно рассматривать как модель, которая определяет информационно-функциональной организации системы (1.1).

Компоненты информационной системы по функциям, которые она выполняет, можно разделить на три слоя: слой отображения, слой бизнес-логики и слой доступа к данным (см. рис. 1.2).

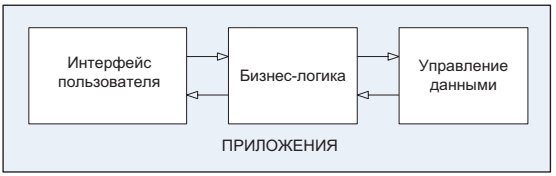


Рис. 1.2 Структура приложений информационных систем

Слой отображения (интерфейса) - все, что связано с взаимодействием с пользователем: нажатие кнопок, движение мыши, рисование изображения, выведения результатов поиска, и тому подобное.

Бизнес-логика - правила, функции, алгоритмы реакции приложения на действия пользователя или на внутренние события, правила и программы обработки данных.

Слой доступа к данным - хранение, выборка, модификация и удаление данных, связанных с решаемой приложением прикладной задачей.

С точки зрения программно-аппаратной реализации компонентов информации системы можно выделить ряд типовых архитектур ИС, которые усовершенствуются с развитием сетевых технологий. Можно выделить пять осно-в них этапов развития (см. рис. 1.3). Сначала на сортировочных станциях использовались много терме-ные системы с распределением времени на базе младших моделей ЕС ЭВМ - ЕС 1010, ЕС 1022, а на сортировочных станциях - системы реального масштаба времени на базе мини-ЭВМ СМ-2М. С появлением микропроцессорных ЭВМ и локаль-ных сетей изначально строились одноранговые сети для обмена файлами, в которых все компьютеры - равноправные, а потом возникла идея выделения отдельной машины для хранения общедоступных файлов. Это были первые 2-х уровневе системы "толстый клиент - файл-сервер" (HC-FS).

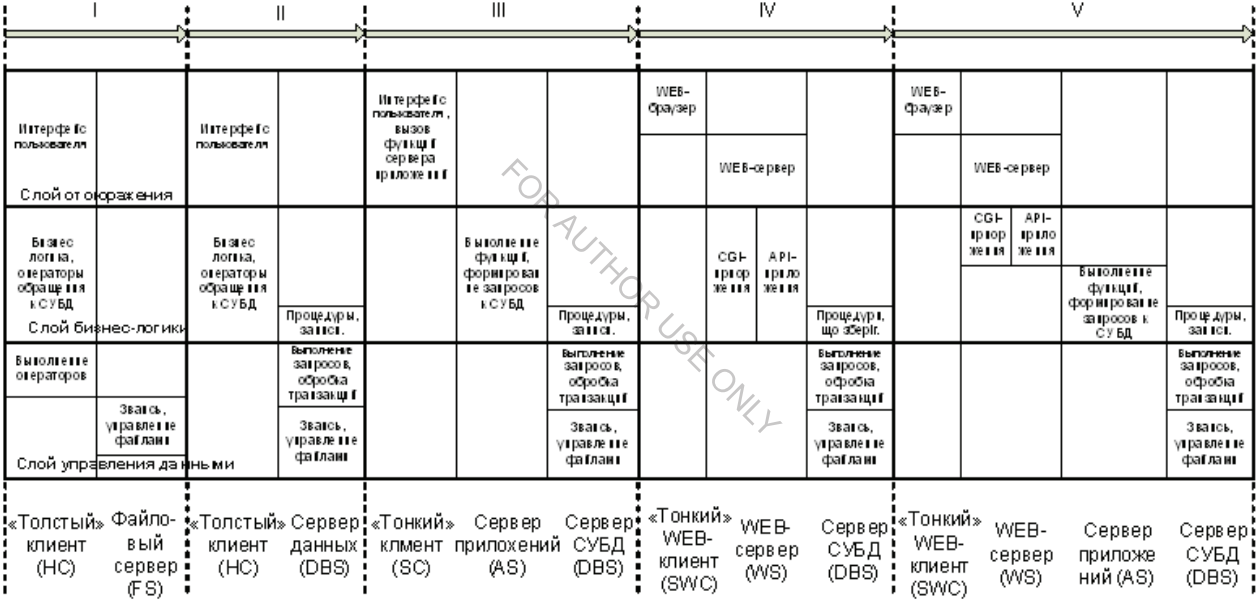


Рис. 1.3 Этапы развития архитектуры

Файл-серверные приложения - приложения, схожие по своей структуре с локальными приложениями и используют сетевой ресурс для хранения программ и данных. Функции сервера: хранение данных и программ. Функции клиента: обработка данных (только на стороне клиента). Количество клиентов ограничено десятками. Преимуществами таких систем стали: многопользовательский режим работы с данными; удобство централизованного управления доступом и низкая стоимость разработки. В то же время выявились и недостатки: низкая производительность и надежность, слабые возможности расширения.

Недостатки архитектуры с файловым сервером вытекают главным образом из того, что данные хранятся в одном месте, а обрабатываются в другом. Это значит, что их нужно передавать по сети, что приводит к очень высоким нагрузкам на сеть и, вследствие этого, резкого снижения производительности приложений при увеличении числа одновременно работающих клиентов. Вторым важным недостатком такой архитектуры является децентрализдванное решения проблем целостности и согласованности данных и одновременного доступа к данным. Такое решение снижает надежность приложений.

Следующий этап развития архитектуры связан с развитием систем управления базами данных (СУБД) и абстрагированием от внутреннего представления данных (физической схемы данных). Теперь клиентские программы манипулируют данными на уровне логической схемы. Использование архитектуры "толстый клиент-сервер БД" (HC-DBS) позволило создавать надежные (в смысле целостности данных) многопользовательские ИС с централизованной базой данных, независимые от аппаратной (а часто и программной) части сервера БД и поддерживающие графический интерфейс пользователя на клиентских станциях, связанных локальной сетью. При этом затраты на разработку приложений существенно сокращались.

Отметим главные особенности этой архитектуры**.** Клиентская часть работает с данными через запросы к серверному ПО. Базовые функции приложения разделены между клиентом и сервером.

Новая архитектура позволила обеспечить полную поддержку многопользовательской работы и гарантию целостности данных. Одновременно стали имеющиеся ее недостатки:

а) бизнес-логика приложений осталась в клиенте, то есть они оставались "толстыми"; при любом изменении алгоритмов, надо обновлять ПО

пользователя на каждом клиентском месте;

в) высокие требования к пропускной способности коммуникационных каналов с сервером, препятствует использование клиентских станций иначе как в локальной сети;

c) слабая защита данных от взлома, особенно от недобросовестных пользователей системы;

d) высокая сложность администрирования и настройки рабочих мест пользователей системы;

e) необходимость использовать мощные ПК на клиентских местах ("толстые" клиенты)

f) высокая сложность разработки системы из-за необходимости использования бизнес-логики и обеспечения интерфейса пользователя в одной программе.

Нетрудно заметить, что большинство недостатков классической или 2-х слойной архитектуры клиент-сервер результате использования клиентской станции в качестве исполнителя бизнес-логики ИС. Поэтому логичным шагом дальнейшей эволюции архитектуры ИС появилась идея "тонкого клиента", то есть разделение алгоритмив обработки данных на части, связанные с отображением информации в удобном для человека представлении, с первичной проверкой данных (клиентская часть) и д "связанные с выполнением бизнес-функций и функций работы с базами данных (серверная часть). Это так называемая промежуточная 2,5-слойная архитектура. Использование хранимых процедур и функций обработки данных бизнес-логики на стороне сервера сокращают трафик, увеличивают безопасность.

Так что такая организация системы весьма напоминает организацию первых много терминальных систем с той лишь разницей, что на месте пользователя стоит не терминал, а персональный компьютер с графическим интерфейсом. Конечно, такое возвращение к почти унитарного системы произошло уже на другом технологическом уровне. Обязательным стало использование СУБД со всеми их преимуществами. Программы для серверной части пишут, в основном, на специализированных языках, пользуясь механизмом хранимых процедур сервера БД. Таким образом, на уровне логической организации, ИС в архитектуре клиент-сервер с тонким клиентом расщепляется на три слоя - слой данных, слой бизнес-функций (хранимые процедуры) и слой отображения. К сожалению, как правило, в такой схеме построения ИС не удается написать всю бизнес-логику приложения на не

предназначенных для этого встроенных языках СУБД. Поэтому, очень часто некоторые бизнес-функции реализуются в клиентской части си-стем, которая от этого неизбежно "толстеет". Потому что физически такие ИС состоят из двух компонентов, эту архитектуру часто называют 2,5-слойный клиент-сервер.

В отличие от 2-х слойной архитектуры 2,5-слойная архитектура обычно не требует наличия высокоскоростных каналов связи между клиентской и серверной частями системы, так как по сети передаются уже готовы результаты вычислений - почти все вычисления проводятся на серверной стороне. Существенно улучшается также и защиту информации - пользователям даются права на доступ к функциям системы, а не на доступ к ее данных и т.д. Однако наряду с достоинствами унитарного подхода архитектура 2,5 перенимает и все его недостатки, то: ограниченную масштабируемость, зависимость от программной платформы, ограниченное использование сетевых вычислительных ресурсов. Кроме того, программы для серверной части системы пишутся на встроенных в СУБД языках описания хранимых процедур, предназначенных для валидации данных и построения несложных отчетов, а вовсе не для написания ИС масштаба предприятия. Все это снижает быстродействие системы, повышает трудоемкость создания и модификации ИС и самым негативным образом сказывается на стоимости аппаратных средств, необходимых для ее функционирования.

Для решения этих проблем и была предложена так называемая 3-х слойная архитектура клиент-сервер (SC-AS-DBS). Основным ее отличием от архитектуры 2,5 является физическое разделение программ, отвечающих за хранение данных (СУБД) от программ, эти данные обрабатывают (сервер приложений, application server (AS)). Такое разделение программных компонентов позволяет оптимизировать нагрузку как на сетевое, так и на вычислительное оборудование комплекса.

Компоненты три звеньевой архитектуры SWC-WS-DBS, с точки зрения программного обеспечения, реализующих определенные браузеры, web-сервера и сервера БД. Место любого из этих компонентов может занять программное обнспечение любого производителя. Например, сервер БД может быть представлен MySQL сервером или MS SQL-сервером; сервер приложений технологиям: ADO.NET, ASP.NET и web-сервером IIS или Apache; роль клиента выполняет любой web-браузер.

Преимущества трехслойной архитектуры: a) тонкий клиент; b) между клиентской программой и сервером приложений передается только

минимально необходимый поток данных - аргументы для функций, вызываемых и значение этих функций, возвращаются; c) сервер приложений может быть запущен в одном или нескольких экземплярах на одном или нескольких компьютерах, позволяет эффективно и безопасно использовать вычислительные мощности организации; d) дешевый трафик между сервером приложений и СУБД; трафик между сервером приложений и СУБД может быть больше, однако это всегда трафик локальной сети, а их пропускная способность достаточно большая и дешевая. В крайнем случае, всегда можно запустить AS и DBS на одной машине, что автоматически сведет сетевой трафик к нулю; e) снижение нагрузки на DBS по сравнению с 2,5-слойной схеме, а значит и повышение скорости работы системы в целом; f) дешевле наращивать функциональность и обновлять ПО.

Переход к 4-хуровневой архитектуре SWC-WS-AS-DBS произошел с развитием интернет-технологий, когда скорости передачи данных в сети стали тождественны со скоростями выполнения многих приложений и существенно увеличилась нагрузка на WEB-сервер обслуживанием запросов, поступающих. В это же время (с 2010) появляются технологии виртуализации вычислений и первые центры обработки данных (ЦОД) - фабрики, предоставляющих информационные услуги (сервисы) с настроенными вычислительными ресурсами для хранения и обработки данных и необходимым для этого программным обеспечением для удаленных пользователей в сети. Впервые идея ЦОДов была сформулирована в середине 80-х годов прошлого столетия фирмой IBM, когда в период повального увлечения серверами, персональными компьютерами и сетевыми структурами они отмечали, что в сети не обойтись без по-тоскующих и высоконадежных центров на базе мэйнфреймов. И ввели термин e-режецентричнои архитектуры сетей и сетецентрической вычислений. В это же время 1981-1991 года в Японии был разработан стратегический компьютер 5-го поколения с интеллектуальным интерфейсом, который был построен на сетях микропроцессоров с мощной базой знаний. Это фактически ЦОД, появившийся на 30 лет раньше своего времени. Законченную в Японии разработку закрыли и признали получены решения неэффективными. И вот через 30-лет, благодаря высокоскоростной Интернет, эта идея воплотилась в ЦОДах.

На рис. 1.4 приведен вариант реализации двух уровней AS-DBS в системе АСК ВП УЗ [38]. На этих уровнях показаны: один из способов увеличения производительности системы (кластеризация AS и DBS), использование сетей с

различными скоростями передачи данных и организация резервирования.

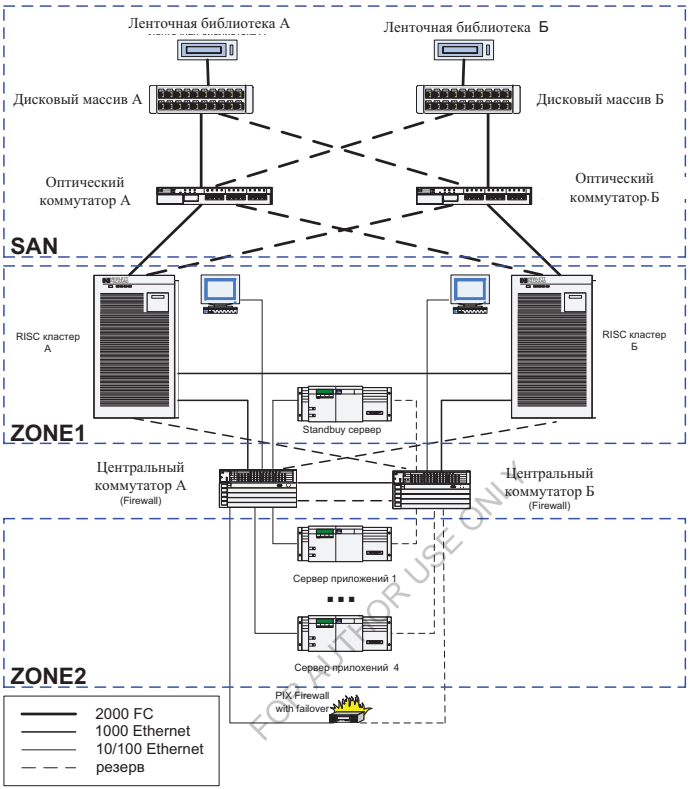


Рис. 1.4 Вариант технической структуры (уровни AS-DBS) АСК ВП УЗ

Указанные этапы развития архитектуры информационных систем в зависимости от развития информационных технологий и пропускной способности сетей передачи данных доказывают, что создаются благоприятные условия для интеграции автоматизированных систем различных уровней. При этом крайне актуальной становится унификация интеграционных решений, так как это позволяет сокращать сроки работ и снижать затраты практически на всех этапах жизненного цикла автоматизированных систем, включая разработку, внедрение и эксплуатацию [55]. В России в 2004 году в Концепции построения архитектуры АСУ РЖД определенные базовые понятия, общие требования и подходы к проектированию бизнес-архитектуры и системной

архитектуры.

Обязательные для исполнения требования к архитектуре впервые были нормативно закреплены распоряжением № 1272р от 21.12.2005 года "Об утверждении общих требований к интеграции и взаимодействия автоматизированных систем". Они предусматривали, в частности, что ИС должны иметь три звеньевую архитектуру, а прикладные функции реализованы на серверах приложений и доступны через их программный интерфейс. Интеграция систем должна осуществляться путем взаимодействия их серверов приложений, прямой доступ к базам данных не допускается. Необходимо построение общей корпоративной шины взаимодействия на основе протоколов, имеющих статус международных, и сервис-ориентированной архитектуры (SOA) [55].

Такая организация взаимодействия и интеграции систем позволяет говорить об их интероперабельность, то есть совместимость на уровне единых протоколов информационного обмена [118].

ОПРЕДЕЛЕНИЕ. Интероперабельность (англ. Interoperability - способность к взаимодействию) - это способность системы, интерфейсы которой полностью открыты, взаимодействовать и работать с другими системами без каких-либо ограничений доступа и реализации [159].

Как отмечается в [55], для повышения технологичности построения общей модели интеграции АСУ РЖД целесообразно использовать готовые инструменты проектирования бизнес-процессов в интеграции со средствами моделирования систем и Реестру автоматизированных систем.

1.4 Развитие архитектуры станционных систем автоматизации в Украине

Автоматизация сортировочных станций начиналась в рамках комплексной программы автоматизации железнодорожного транспорта (АСУЖТ) на основе электронных вычислительных машин ЕС ЭВМ. В конце 1970-х - начале 1980-х годов появляются первые модели, отражающие формирование, движение и расформирования поездов. Параллельно появляется модель сортировочной станции - основа автоматизированной системы управления сортировочной станцией и первая система АСУ СС (1975 год). В системе впервые была реализована выдача плана формирования ТГНЛ и сортировочного листа на основе автоматизированного оформления ТГНЛ и базы данных назначений

плана формирования. Особо следует отметить среди автоматизированных функций ведения модели накопления вагонов на путях сортировочного парка, формирование состава по накоплению вагонов, выдачу на сформированный поезд натурного листа и справки маршрута машиниста. Первые АСУ СС строились на ЕС 1010 - младшей из моделей ЕС ЭВМ. В Украине системы были запущены в эксплуатацию на станциях Дарница и Купянск-Сортировочный. В дальнейшем система модернизировалась путем замены ЕС 1010 на ЕС 1022.

Развитие программно-технических средств, связанный с внедрением мощных моделей фирмы IBM, активизировал создание вагонных моделей и моделей поездов сетевого уровня и в 1980-х годах появляется автоматизированная система оперативного управления перевозками (АСОУП) и на ее основе - системы автоматизированного диспетчерского центра управления (АДЦУ). АСОУП, как система дорожного уровня, обеспечила автоматизацию информационного обмена между АСУ СС станций.

С образованием независимых государств работы, связанные с автоматизацией сортировочных станций, в России, Беларуси и Украины пошли по своим независимыми направлениям. В России разрабатываются системы АСК СТ [11], АИСТ [137], АСТРА СС [130], в Беларуси - система АГАТ [50], в Украине - появляется Комплексная система электронного обмена документами (КСЕОД-СС), разработанная специалистами ИВЦ ЮЖД дороги с участием инженеров программистов Миронова И.Н., Хатунцев К.В., а также специалистов ИВЦ Львовской и Юго-Западной железных дорог. Функционально система КСЕОД-СС была эквивалентна АСК СС, но благодаря новым техническим средствам - персональным компьютерам - была более эффективной по быстродействию, интерфейсом, энергопотреблением. Систему можно было разворачивать на серверах любых уровней. Имея широкие функциональные возможности система КСЕОД-СС была рекомендована для внедрения на железных дорогах Украины до сих пор работает на некоторых решающих станциях.

С появлением системы АСУ ГП УЗ с единым центром обработки данных появилась новая автоматизированная подсистема в составе АСУ ГП УЗ - "Динамическая работа станционного узла" (разработчик ПКТБ АСУЖТ Украины). Она позволила устранить недостатки предыдущих версий системы: последовательная обработка информационных потоков от АРМ-ов станции - сначала в АСУ СС, потом в АСУ ГП УЗ, что ухудшало время реакции системы; отсутствие данных по вагонам и данных с перевозочных документов. К концу

2011 года система внедрена на всех сортировочных станциях страны [34]. Этапы развития систем автоматизации сортировочных станций представлены на рис. 1.5.

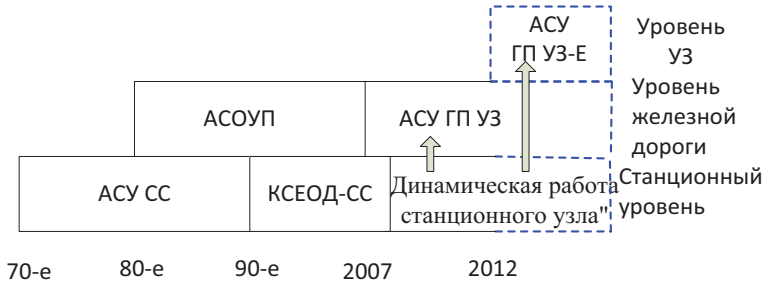


Рис. 1.5 Этапы развития систем автоматизации сортировочных станций

1.5 АСУ ГП УЗ-Е как основа интеграции информационно-управляющих систем жнлезгых дорог Украины

Новые ресурсосберегающие системы автоматизации сортировочных станций должны интегрироваться в единую систему на основе АСУ ГП УЗ-Е. Поэтому необходимо исследовать, какие ресурсные резервы имеет эта система.

1.5.1 Четырехуровневая структура построения системы

Для исследования характеристик АСУ ГП УЗ-Е будем использовать полную 4-х уровневую модель системы, представленную на рис. 1.6.

Комплексирования вычислительных ресурсов на разных уровнях осуществляется либо путем построения многомашинных систем, или путем использования многопроцессорных систем (симметричное мультипроцессирование - SMP). Эффективность этих вариантов определяется количеством процессоров, которые используются. На рис. 1.7 (по данным www.tpc.org) показана производительность систем на смеси (бенчмарке) TPC-C (в количестве транзакций в минуту - tpmC), приходящаяся на один процессор [204]. TPC-C предполагает, что ≥ 90% транзакций имеют ограничения на время ответа (в среднем 5 секунд).

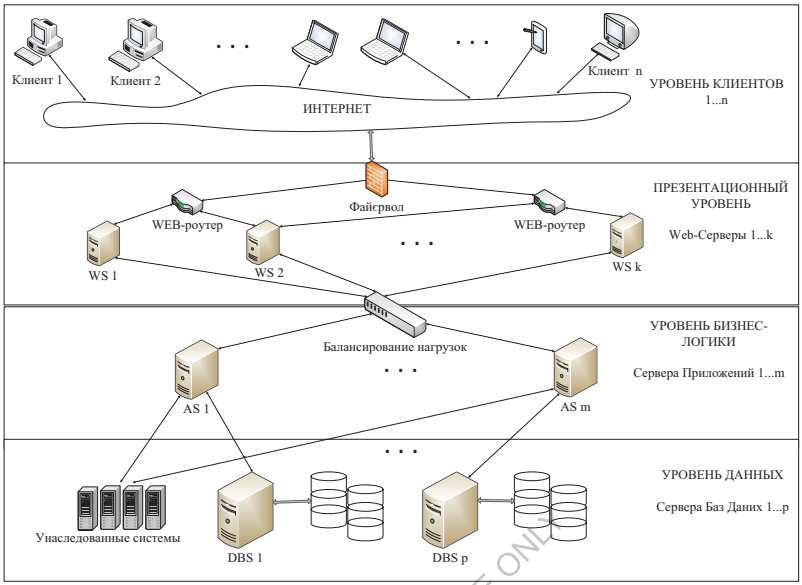


Рис. 1.6 Четырехуровневая структура АСУ ГП УЗ-Е

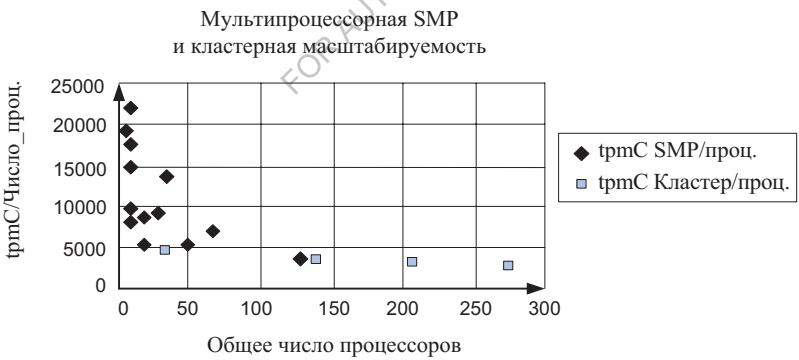


Рис. 1.7 SMP и кластерная масштабируемость

Таким образом, для системы АСУ ГП УЗ-Е можно рекомендовать при большом количестве серверов использовать кластеризацию (например, на уровнях WS, AS) и технологию SMP при количестве серверов - до 15 (уровень DBS).

В недолгой истории развития компьютерных систем можно четко отделить ряд основных этапов, которые характеризуются преимущественной архитектурой построения таких систем (см. рис .1.3).

1 - HC-FS;

2 - HC-DBS;

3 - SC-AS-DBS;

4 - SWC-WS-DBS;

5 - архитектура с четырьмя и более уровнями (SWC-WS-AS-DBS).

Этот перечень архитектур можно дополнить еще двумя:

6 - одноранговая (пиринговая, p2p) архитектура;

7 - сервис-ориентированная архитектура;

8 - сетецентрическая сервис-ориентированная архитектура.

Каждая новая архитектура появляется в результате повышения сложности информационных систем: размер хранимой информации ежегодно удваивается, число пользователей систем увеличивается в геометрической прогрессии, появляются различные сервисы, которые работают на разных платформах и т.д. Современным решением является сетецентрический подход к проектированию информационных систем, в основе которого лежит сервис-ориентированная архитектура.

Сервис-ориентированная архитектура (service-oriented architecture, SOA) - модульный подход к разработке информационных систем, в основе которого лежить разработка сервисов со стандартизированными интерфейсами [43, 55].

Отметим только одну основную характеристику сетецентрической системы, которая важна для ИС реального времени. Это скорость принятия решений (speed of command) - время, необходимое для прохождения полного цикла Бойда "Наблюдение - Ориентация - Решение - Действие" (Observe - Orient - Decide - Act, OODA) [43]. Такой цикл включает в себя: сбор информации из внутренних и внешних источников - наблюдение; формирование множества возможных вариантов и оценка каждого из них по совокупности критериев - ориентация; выбор лучшего плана действий для практической реализации - решения; практическая реализация выбранного плана действий - действие. Очевидно, что количество уровней иерархии или этапов в принятии решения по OODA не должно превышать четыре. Основу таких систем составляют Центры обработки данных (ЦОД) [127, 141].

В настоящее время АСУ ГП УЗ-Е можно представить как сетецентрическую, сервис-ориентированную, 4-х уровневую информационную

систему реального времени, модель которой показана на рис. 1.8. Ее можно рассматривать как переходную к "облачных вычислениям" - динамичным ИТ-сервисам на базе WEB [51, 155].

Рассмотрим теперь первичные данные для моделирования АСУ ГП УЗ-Е.

1.5.2 Вычислительные и коммуникационные ресурсы АСУ ГП УЗ-Е

Система АСУ ГП УЗ-Е реализована на базе модели P780 фирмы ИБМ [16], которая имеет следующие основные характеристики (максимальная конфигурация): восемь 3.86 GHz POWER7 восьмиядерных процессорных модулей; 64 ядра 3.86 GHz 256 KB L2 cache на ядро; 8 MB L3 cache на ядро (eDRAM); до 1 TB 1066 MHz DDR3 или до 2 TB 800 MHz DDR3; Active Memory Expansion; до 24 дисков SFF или до 24 дисков SFF SAS, четыре slimline для SATA DVD-RAM; 24 PCI Express x8; операционные системы AIX, SLES, RHEL.

В моделях максимальный вычислительный ресурс системы составляет 64 процессора и 24 обычных SATA или SAS-дисков (Serial Attached SCSI). SAS-диски имеют более высокие скорости выполнения дисковых операций ввода-вывода.

Коммуникационные ресурсы системы имеют следующие характеристики.

Скорости внутренних локальных сетей в пределах верхних уровней AS-DBS: 10/100 Мбит /с, 1 Гбит /с, 2 Гбит /с.

Что касается внешних каналов, то в настоящее время на железных дорогах Украины используются 2-х уровневые волоконно-оптические линии связи (ВОЛС): магистральный уровень (STM-16) с максимальной пропускной способностью до 2,5 Гбит/с; дорожный уровень (STM-4) с максимальной пропускной способностью до 622 Мбит/с. Для системы АСУ ГП УЗ выделялась пропускная способность 10 Мбит/с [52]. Но в узлах МПД, где нет ВОЛС, используются выделенные и коммутируемые каналы тональной частоты, каналы мобильной и спутниковой связи (GRPS, EDGE). Средняя пропускная способность этих каналов составляет от 19, 2 до 56 Кбит/с, чего явно недостаточно для работы в АСУ ГП УЗ-Е.

Внешние каналы передачи данных в новой системе АСУ ГП УЗ-Е - 50 Мбит/с с дорогами, на уровне дорог каналы закольцованы [10]. Приведенные выше данные говорят, что существуют достаточные резервы повышения пропускной способности внешних каналов. Поэтому в дальнейшем остановимся только на нужных вычислительных ресурсах.

1.5.3 Особенности построения и функционирования системы

Разработка основных проектных решений по созданию единой АСУ грузовыми перевозками осуществлялась исходя из двух основных положений [147]:

- сохранение общей методологии (идеологии) АСУ ГП УЗ-Е;
- построение системы с единым централизованным серверным узлом.

Одним из основных принципов построения АСУ ГП УЗ-Е является декомпозиция системы на множество технических, программных и информационных компонентов (см. определение понятия архитектура в данной работе - (1.1)) с максимальной стандартизацией взаимодействия всех компонентов, обеспечивает:

- максимальную открытость системы к наращиванию функциональных возможностей (появление новых задач и модификация существующих);
- простоту сопровождения и развития системы;
- возможность распараллеливания разработки.

Функциональные проектные решения в системе объеденены в функционаные комплексы (ФК): ФК организации перевозочного процесса; ФК коммерческой работы; ФК вагонного хозяйства; ФК локомотивного хозяйства.

1.5.4 "Предельные" проектные оценки характеристик системы

Все проектные решения разрабатывались исходя из таких оценок размерностей системы, которые можно использовать для оценки "предельных" характеристик системы:

количество железнодорожных абонентов, участвующих в формировании БД - до 20000; они обеспечивают среднесуточное обновления базы данных - до 500 000 (нагрузка на AS-DBS: 5,79 транзакций/с; принимаем $t_{\text{обр}}^{\text{AS}} = 0,2\,\text{с}$ $t_{\text{обр}}^{\text{DBS}} = 0,6\,\text{с}$);

количество железнодорожных абонентов, пользующихся информационно-справочной системой - до 25000; среднесуточное количество отчетных документов, выдаваемых (справок) - до 1 миллиона (нагрузка на AS-DBS: 11,58 транзакций/с; будем предполагать, что $t_{\text{обр}}^{\text{AS}} = 0,1\,\text{с}$ $t_{\text{обр}}^{\text{DBS}} = 0,3\,\text{с}$);

количество взаимодействующих АСУ клиентов - до 10000; среднесуточное количество транзакций - 60 000 (нагрузка на AS 0,69 транзакций/с; принимаем

$t_{\text{обр}}^{\text{AS}} = 0,2\,\text{с}$).

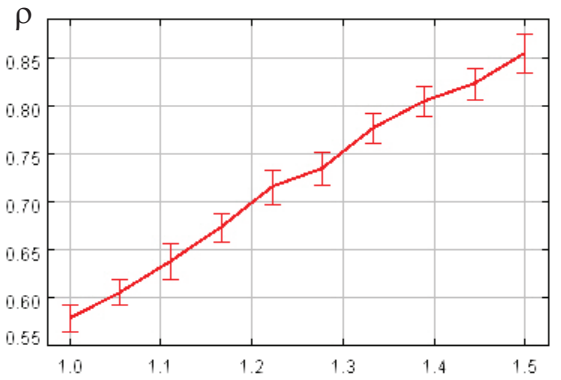
Определим предельные характеристики системы АСУК ГП УЗ-Е на основе введенных оценок. Временные характеристики обработки транзакций, приведенные выше, принятые по данным [10, 124, 127, 141]. Расчетное количество процессоров в кластерах по уровням - $K^{\text{WS}} = 1;\ K^{\text{AS}} = 6;\ K^{\text{DBS}} = 12.$

В качестве начального уровня входных потоков примем те, которые были использованы в проекте (на рис. 1.8-1.13 это соответствует значению 1). При этом 6 серверов в кластере AS обеспечивают 50% запас по средней загрузки (см. рис.1.8), а вот в кластере DBS 12 серверов обеспечивают всего 5% запас до уровня (см. рис.1.9).

Рис. 1.8 Средняя загрузка кластера AS

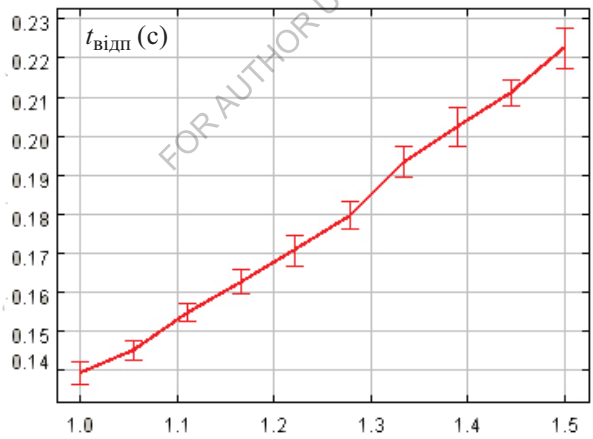
Поскольку АСУ ГП УЗ-Е рассматривается как система реального масштаба времени, то весьма важным параметром может быть время отклика системы. На рис. 1.10 и 1.11 показана зависимость времени реакции от входных потоков. Если для серверов приложений она имеет практически линейный вид, и есть достаточный запас в приемлемом диапазоне времени (см. рис. 1.10), то для серверов баз данных кривая имеет экспоненциальный вид и при 50% нагрузке (относительно базового уровня) достигает критических значений (см. рис. 1.11). Отметим, что это в том случае, если будут увеличены проектные "предельные" параметры размерности системы [147]. Таким образом, нужны "проектные" вычислительные ресурсы (19 процессоров) с избытком

перекрываются ресурсами системы P780 (64 процессора).



DBS
K=12

Рис. 1.9 Средняя загрузка кластера DBS



AS
K=6

Рис. 1.10 Среднее время ответа кластера AS

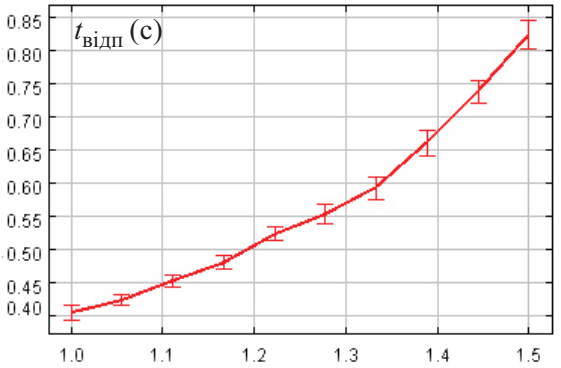


Рис. 1.11 Среднее время ответа кластера DBS

1.5.5 Оценка эксплуатационных характеристик АСУ ГП УЗ-Е

Система АСУ ГП УЗ-Е введена в промышленную эксплуатацию 7 июля 2012 и уже известны первые результаты ее эксплуатации, которые можно использовать для моделирования и исследования системы [10, 124, 127, 141]. При этом входные потоки несколько отличаются от их проектных отметок [147].

В настоящее время с системой работают 25 000 пользователей из 12 хозяйств железных дорог Украины, которые формируют входной поток запросов от 650 тысяч до 1 млн. в сутки, что соответствует интенсивности 7,5 - 11,6 транзакций/с. Потоки в системе распределяются следующим образом: на первых двух уровнениях WS/AS - 0,3/0,7; дальше, на AS/DBS - 0,45/0,55.

Среднее время обработки заявок на процессорах AS-DBS распределим пропорционально сложности операций в соответствующих серверах $t_{\text{обр}} = t_{\text{AS}} + t_{\text{DBS}} = 0,2 + 0,6 = 0,8\text{с}$.

Средний размер сообщений, поступающих в систему, примем 21 Кбайт, распределение размеров сообщений - по закону Парето [204].

В данной работе рассмотрим макромодель системы, которая позволяет определить нужные вычислительные ресурсы по уровням WS-AS-DBS с учетом развития АСУ ГП УЗ-Е (увеличение входного потока заявок, осложнение и увеличение количества приложений в системе). Выполненные исследования на

аналитических моделях M/M/K и Par/M/K [101] дальше следующие приближенные результаты (без структуризации запросов, без оценки сложности приложений и учитыванием особенностей работы с дисковыми массивами).

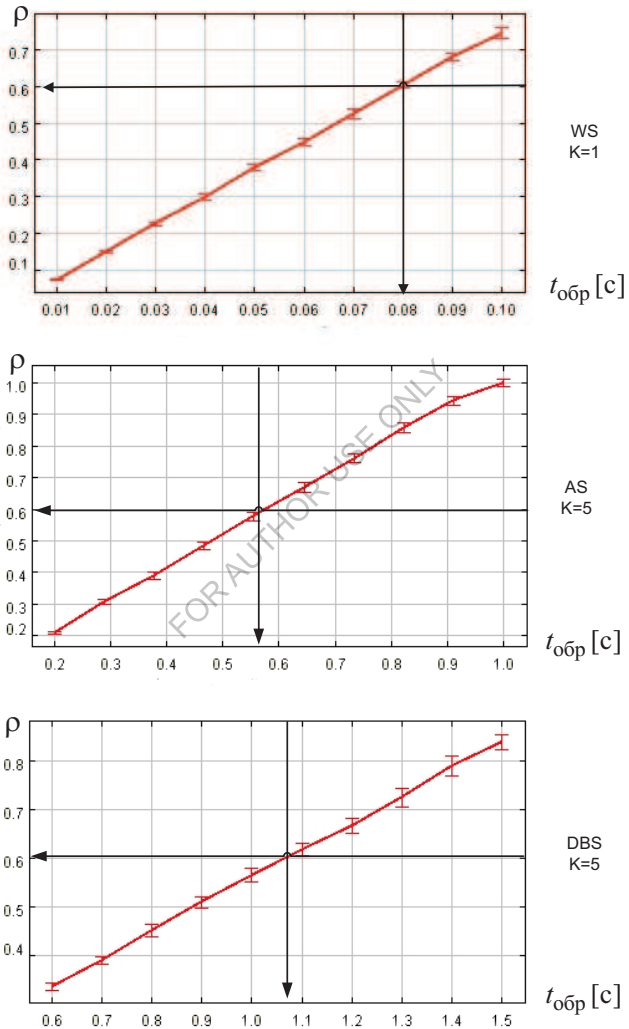


Рис. 1.12 Средние загрузки процессоров по кластерам системы в зависимости от времени обработки запросов

При моделировании предусматривались следующие базовые параметры обработки запросов: $t_{\text{обр}}^{\text{WS}} = 10$ мс; $t_{\text{обр}}^{\text{AS}} = 200$ мс; $t_{\text{обр}}^{\text{DBS}} = 600$ мс. При этом количество процессоров в кластерах по уровням $K^{\text{WS}} = 1$; $K^{\text{AS}} = 5$; $K^{\text{DBS}} = 5$. На графиках показаны резервы по усложнению задач обработки запросов до уровня, когда загрузка достигает 0,6 (см. рис. 1.12).

В этом случае следует расширять кластеры, для чего есть все необходимые резервы в системе P780.

Применение распределения Парето для внешних заявок не влияет на характеристики уровней AS-DBS, так как сглаживается на уровне WS. Поэтому в дальнейших исследованиях можно пользоваться экспоненциальным распределением и моделями M/M/K.

Для системы реального времени среднее время ответа может быть одной из главных характеристик, для которой устанавливаются определенные требования заказчика. В этом случае можно воспользоваться зависимостями, вид которых отображено на рис.1.13.

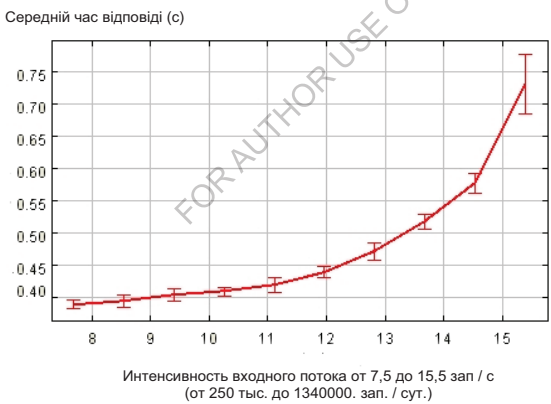


Рис. 1.13 Зависимость среднего времени ответа системы от входного потока запросов

Полученные результаты показывают, что система АСУ ГП УЗ-Е в процессе первых месяцев эксплуатации использует только 60% от проектных ресурсов и 17% от вычислительных ресурсов системы P780. Таким образом, есть все необходимые предпосылки для ее расширения задачами планирования и управления процессами расформирования-формирования поездов на сортировочных станциях.

2. Развитие комплексного подхода к системному анализу, проектированию и совершенствованию систем автоматизации

2.1 Особенности проектирования и эксплуатации сложно структурированных систем управления реального времени

Процесс проектирования устройств автоматизации всегда начинается с анализа системы управления, для которой они создаются. Исследованию подлежат такие ее характеристики как количество контролируемых параметров, время реакции управляемого объекта на управляющие воздействия, требуемая точность управления. Кроме них важное значение имеет математическая модель функционирования системы - совокупность соотношений, определяющих зависимости изменения регулируемых величин от входных воздействий, формируемых вычислительным комплексом.

Функционирование системы можно представить вектор-функцией $\mathbf{S}(t)$, значение которой в моменты времени $t_0,...,t_i,t_{i+1},...$ определяют состояния системы $\mathbf{S}(t_0),...,\mathbf{S}(t_i),\mathbf{S}(t_{i+1}),...$. Каждое состояние характеризуется множеством величин, описывающих внешние факторы (они называются действиями), влияющие на систему, и протекания процессов внутри самой системы.

Назначение системы управления - целенаправленное изменение состояния объекта управления (подвижных отцепов), или, что то же самое, управление процессом функционирования объекта - сортировочной горки. Чтобы управлять, необходимо знать: как ведут себя объекты управления (состояния отцепов в заданные моменты времени); какие те внешние воздействия на отцепы, которыми мы не можем управлять (действия внешней среды); какова цель управления; какие средства воздействия на объект мы имеем (какие ресурсы имеются в нашем распоряжении - замедлители, стрелочные переводы и т.п.).

На рис. 2.1 показаны взаимосвязи объекта управления - сортировочной горки, системы управления и окружающей среды. Действия, производимые системой управления будем называть *управляющими действиями*. На рис. 2.1 они обозначены $\mathbf{X}=(x_1,x_2,\ldots,x_k)$. Действия, не зависящие, от управляющей системы, будем называть *возмущениями*. Часть их измеряется (контролируемые возмущения) $\mathbf{Z}=(z_1,z_2,\ldots,z_o)$, часть по разным причинам не измеряется

(неконтролируемые возмущения) $\mathbf{F} = (f_1, f_2, \ldots, f_n)$, причем точное число неконтролируемых возмущений обычно неизвестно. Контролируемые величины, по которым ведется управление, будем называть *управляемыми* или *регулируемыми*. Они обозначены $\mathbf{Y} = (y_1, y_2, \ldots, y_r)$.

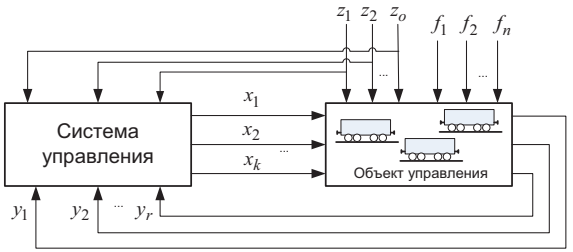


Рис. 2.1 Взаимосвязи объекта управления, системы управления и окружающей среды

В системах управления сортировочными горками реализуются, как правило, наиболее сложные задачи адаптивного управления, в которых управляющие действия, или алгоритм управления, автоматически и целенаправленно меняется для осуществления в каком-либо смысле наилучшего управления объектом, причем характеристики объекта или действия внешней среды могут изменяться заранее непредсказуемым образом.

Во всех случаях управляющая система определяет необходимые управляющие действия в соответствии с определенным законом управления $\mathbf{X}(t) = \Phi(\mathbf{Z}, \mathbf{Y}, t)$. Для реализации сложных законов управления нужны сложные алгоритмы и программные комплексы, отличием которых является наличие жестких ограничений на время решения задач управления, что обусловлено высокой скоростью изменения возмущений $\mathbf{F}(t)$, действующих на объект управления.

Практика показывает, что через упомянутые выше особенности, а также из-за наличия жестких требований и ограничений, проектирование информационно управляющих комплексов для автоматизации сортировочных горок является наиболее сложной инженерной задачей.

Различия в характере и сложности управляемых объектов (отцепов) не могут не сказываться на требованиях к характеристикам и структуры АСУ, что, в свою очередь, влияет на выбор параметров и методику проектирования вычислительных комплексов, входящих в ее состав. Для общей оценки этого

влияния выделим важнейшие показатели и рассмотрим их особенности.

Количество контролируемых параметров. К ним относятся входные воздействия $\mathbf{X}(t)$ і $\mathbf{Z}(t)$, регулируемые величины $\mathbf{Y}(t)$, а также некоторые параметры, описывающие состояние объекта управления, причем часть их не меняется в процессе управления или меняется очень редко (практически постоянные параметры). Как было определено в в первой главе, $\mathbf{Z}(t)+\mathbf{Y}(t)$ является условной информационной мощностью (УИМ) объекта управления. По действующим стандартам сортировочные горки в зависимости от объекта управления (количества пучков на горке) относятся к системам повышенной и большой (максимальной) УИМ с количеством входных сигналов от 809 до 2153.

С ростом сложности объекта управления или условий внешней среды, в которых функционирует объект, количество контролируемых параметров увеличивается. С ростом их количества возрастает сложность принятия решения по управлению, поскольку увеличивается число факторов, которые учитываются и резко повышается количество взаимосвязей между ними и регулируемыми величинами. Это приводит к росту числа и сложности алгоритмов управления; усложняется деятельность оператора по анализу обстоятельств и принятия решения; повышаются требования к скорости технических средств, которые собирают и обрабатывают информацию о контролируемых параметрах, формируют сигналы управления; повышаются требования к емкости запоминающих устройств для хранения необходимых в процессе управления величин.

Для обработки информации создаются сложные алгоритмы и комплексы программ, причем их состав в процессе эксплуатации практически не меняется и последовательность решения отдельных задач жестко задана. Поэтому, в дальнейшем будем говорить про φ - транзакции как последовательность задач формирования $x(t_i) \in \mathbf{X}(t)$ (см. раздел 4).

Зависимость регулируемых величин от входных воздействий. Характер и математическая формулировка этой зависимости определяют сложность алгоритмов и программ управления. В АСУ сортировочной горки связь между переменными, описывающие управляемый движение отцепов, задается дифференциальными уравнениями, с параметрами, зависящими от входных и выходных величин.

Динамические характеристики и временные ограничения. Они

определяются скоростью изменения состояния объекта управления (движения отцепов) $\mathbf{S}_{\text{вчп}}(t)$ и зависят от его инерционности и скорости изменения возмущений.

Пусть t_i і t_{i+1} - два момента времени, в которые замеряны возмущение $\mathbf{Z}(t_i)$ і $\mathbf{Z}(t_{i+1})$, требующие выдачи соответствующих управляющих действий. За время $t_{i+1} - t_i$ должна быть обработана необходимая информация, принято решение по управлению, выданны управляющие сигналы, начат и закончен переходный процесс изменения состояния от $\mathbf{S}_{\text{вчп}}(t_i)$ к рассчитанному в процессе принятия решения состояния $\mathbf{S}_{\text{вчп}}(t)$, де $t_i < t \le t_{i+1}$. В зависимости от постановки задачи оптимального управления длина интервала $t - t_i$ часто должна удовлетворять определенные требования (например, может быть поставлена задача ее минимизации).

В этой последовательности действий выделяются две составляющие: принятие решения управляющей системой (время $t_{\text{пр}}$), и отработки принятого решения объектом управления (время его реакции $t_{\text{р}}$). Очевидно, что $t_{\text{пр}} + t_{\text{р}} \le t - t_i$. Если объект управления многосвязный (управление несколькими отцепами), то векторы **X**, **Y** і **Z** имеют несколько координат, то есть это соотношение должно выполняться для каждой координаты. С ростом скорости изменения возмущений длина интервала $t - t_i = \Delta t$ уменьшается, что при неизменных динамических характеристиках объекта приводит к уменьшению допустимого времени $t_{\text{пр}}$, отведенного для принятия решения, поскольку

$$0 < t_{\text{пр}} \le \Delta t - t_{\text{р}}. \qquad (2.1)$$

Это, в свою очередь, вызывает повышение требований к быстродействию технических средств сбора, обработки и выдачи информации и требований к надежности функционирования системы, поскольку затраты времени на восстановление могут привести к нарушению условия (2.1) и изменения степени автоматизированного управления (в автоматизированных системах).

В АСУ сортировочной горкой вагонные замедлители верхней и групповой позиций должны иметь время отпуска не более 0,8 с, а время переведения стрелки, включая время срабатывания коммутирующей аппаратуры, должо быть не более 0,8 с [14]. Для отдельных устройств эти значения могут быть жесткими. Так, например, для замедлителей РНЗ-2М время срабатывания при растормаживании (от подачи команды до полного снятия усилия) не более

0,6 с, а при переходе от расторможенного положения в заторможено - не более 0,7 с [6].

С (2.1) видно, что время, которое отводится управляющей системе для выполнения расчетов по программам, реализующих алгоритм управления $t_{\text{ра}}$, должен удовлетворять условию:

$$t_{\text{ра}} \le t_{\text{пр}} < \Delta t - t_{\text{р}}.$$

Если возмущение меняются быстро (Δt малое), а задачи управления сложные ($t_{\text{ра}}$ большое), то величина $t_{\text{ра}}$ отличается от $t_{\text{пр}}$ на малую величину ε, которая значительно меньше $t_{\text{ра}}$, причем при реализации алгоритма сравнимый со временем переходного процесса объекта управления:

$$t_{\text{ра}} = t_{\text{пр}} - \varepsilon,\ \varepsilon \ll t_{\text{ра}},\ t_{\text{ра}} \approx t_{\text{р}}. \qquad (2.2)$$

В большинстве случаев практическая реализация равенства в соотношениях (2.2) связана с высокими требованиями к быстродействию технических средств автоматизации.

Величина Δt может определяться скоростью изменения не только возмущений, но и других параметров, характеризующих состояние объекта (например, пространственных координат), а также заданными моментами реального астрономического времени.

Из соотношений (2.2) видно, что основным параметром, определяющим скорости всех процессов в системе управления, является величина $t_{\text{р}}$ - время переходного процесса в объекте управления. Она определяет, какой масштаб времени следует выбрать при анализе и синтезе системы.

Соотношение (2.2) показывает, что результаты используются для управления практически немедленно (через малый интервал ε). Управляющая система в этом случае работает как бы в том же масштабе времени, что и объект управления, то есть в реальном времени.

Понятие масштаба реального времени, характеризующий особенности функционирования управляющей системы, часто распространяют и на систему управления в целом. Говорят, что АСУ работает в масштабе реального времени (или просто «АСУ реального времени»), если хотят подчеркнуть ее следующие особенности:

1) чрезвычайно малое время, отведенное для принятия решения, совместимый со временем переходного процесса в объекте;

2) сложность алгоритмов решения задач управления и практически

немедленное использование результатов решения для управления;

3) недопустимости как преждевременной выдачи управляющих сигналов, так и их запаздывания.

Точность управления. При проектировании АСУ этот показатель оценивается как степень отклонения результатов измерений или величин управляющих воздействий от некоторых значений, признанных идеальными при определенных допущениях (идеальный алгоритм вычислений, абсолютная точность представления исходных данных и результатов и т.п.) в данный момент времени. Точность управления также характеризуется степенью отклонения фактических моментов выдачи управляющих воздействий от оптимальных или заданных.

С повышением требуемой точности усложняются алгоритмы решения задач управления (например, из-за увеличения степени полинома, аппроксимирующего характеристику датчика); увеличивается количество разрядов, представляющих информацию, которая передается или обрабатывается; повышаются требования к быстродействию вычислительных комплексов и к скорости передачи информации.

Точность, как параметр, влияющий на выбор технических решений, должны учитываться и при проектировании АСУ технологическими процессами на сортировочных станциях. Нарушение требуемой точности может привести к аварии объекта управления и поэтому расценивается как отказ системы.

Распределенность системы в пространстве и условия протекания управляемого процесса. Управляемый процесс протекает в пространстве и во времени. Изменение координат точек приема и выдачи информации может быть связана как с перемещением самого объекта управления, так и с последовательным выполнением цепочки технологических операций на различном оборудовании. АСУ сортировочных станций является распределенными системами, поэтому важное значение имеет система передачи данных.

Необходимость подхода к анализу и проектированию информационно-управляющих систем на сортировочных станциях как к системам реального масштаба времени указывалась в работах [105, 49, 44, 131], где решались отдельные задачи, связанные с алгоритмизацией процессов управления и разработкой локальных подсистем управления.

Обобщая все рассмотренные положения и анализируя их с точки зрения

требований к вычислительным комплексам, можно сделать следующие выводы.

1. Все технические средства автоматизации, работающих в составе АСУ, ориентированные на реализацию определенных алгоритмов, которые требуют тщательного анализа и оценки времени их выполнения.

2. АСУ сортировочных станций относятся к классу систем реального масштаба времени и характеризуются:

жесткими требованиями к времени формирования решений, величина которого близка к времени реализации алгоритма, что делает необходимым не только оценивать время выполнения алгоритмов, но и другие потери времени в очередях на обслуживание с приоритетами в каналах обработки и передачи данных;

сложными программными комплексами для решения определенных, заранее заданных задач и большим объемом обрабатываемых данных, которые необходимо учитывать при проектировании АСУ;

связи с автоматическими источниками и приемниками информации, в распределенных системах актуальным делает задачу разработки рациональной информационной структуры системы;

высокими требованиями к надежности работы, к достоверности надо учитывать при разработке технических структур и программно-диагностического обеспечения АСУ;

ограничениями на потребляемую энергию и габариты, что определяется при выполнении функционально-логического и конструктивного компоновки унифицированных технических средств автоматизации.

Перечисленные требования следует учитывать в методике проектирования и усовершенствования технических средств автоматизации сортировочных станций. И одной из главных задач является поиск моделей, которые позволяют анализировать дискретные системы реального масштаба времени управления сортировочными горками и формировать принципы декомпозиции-композиции системы управления в условиях временных ограничений (2.2). В качестве таких моделей в работе предлагаются многосортные алгебраические модели дискретных систем реального времени.

2.2 Современные тенденции развития процессов информатизации и их влияние на проектирование информационных систем

Разработка и внедрение современных информационных технологий и

систем во всех отраслях промышленности и транспорта происходят в условиях глобальных структурных социально-экономических, технологических и социальных изменений. К основным факторам, которые влияют на процессы информатизации, а также на проектирование автоматизированных систем и их архитектуру можно отнести:

1 - **тенденции глобализации** социально-экономических отношений, приводящих к образованию транснациональных корпораций с территориально распределенными предприятиями, единым центром управления с соответствующей информационной инфраструктурой, потоками информации и широким кругом задач автоматизации управления корпорациями;

2 - **интеграция информационных систем** в экономических зонах, создаваемых с четкой экономической независимостью каждого участника сотрудничества и стремлением к достижению максимальной экономической эффективности совместной деятельности;

3 - территориальная рассредоточенность объектов управления и информационного обеспечения привели к широкому внедрению интернет-технологий, которые характеризуются **многообразием вариантов программно-технических решений по организации доступа к техническим, программным, информационным ресурсам корпоративных систем;**

4 - открытость ИНТЕРНЕТ-технологий и широкая разветвленность глобальных коммуникаций делает актуальной **задачу обеспечения надежности, живучести и информационной безопасности в процессе разработки корпоративных информационных систем;**

5 - интеграция задач измерения, контроля, управления и информационного обеспечения в современных системах привела к унификации методологий их проектирования, которая заключается в том, что рассматриваются общие **модели и методы проектирования информационных систем (без выделения, как было раньше, измерительных автоматизированных систем, систем контроля, управляющих систем и, тем более, с дополнением - вычислительных систем);**

6 - интеграция различных систем, технологий и задач управления, разрабатывались различными организациями и разработчиками в разное время обнаружила одну из острейших проблем интеграции: **многократное дублирование в информационном обеспечении объединяемых систем (массивов, таблиц и других информационных объектов), при их временной и семантической несогласованности; отсюда - дублирование**

информационных потоков, «информационное засорения» каналов и баз данных, а в результате - проблема недостоверности данных и снижения эффективности управления;

7 - все выше перечисленные факторы приводят к **экспоненциальному росту сложности** создаваемых корпоративных информационных систем, в свою очередь приводит к такому осложнению процессов проектирования, что они переходят к классу «транс вычислительных задач» [57], эффективно решать которые можно только с применением моделей и методов искусственного интеллекта.

W. L. Chapman, J. Rozenblit i A. T. Bahill доказали [176], что системное проектирование АСУ высокой сложности, к которым относятся АСУ сортировочных станций и горок, является NP-полной задачей много вариантного выбора итерационного типа, связанной с обработкой большого объема данных (см. рис. 2.3).

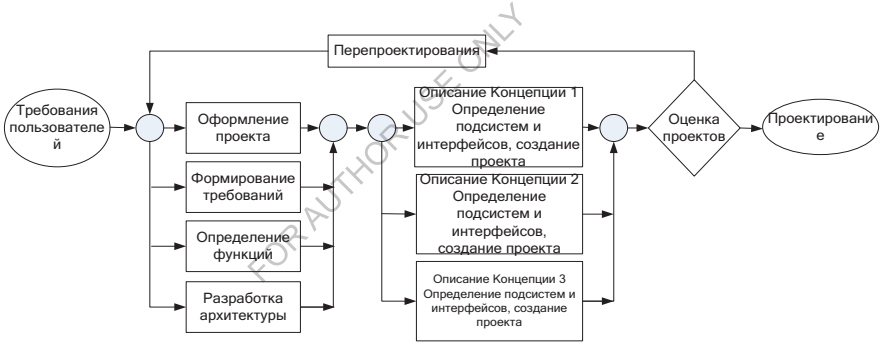


Рис. 2.2 Процесс проектирования систем автоматизации

Создание высокоэффективных интегрированных автоматизированных систем управления, таких как АСУ ГП УЗ-Е, которые объединяют функционально и объект-ноориентованые информационно-управляющие вычислительные системы и сети, практически невозможно без использования автоматизированных систем анализа, проектирования и обучения, которые включают научную методологию , программные и технические средства проектирования и получили название инженерных технологий и инструментальных систем автоматизированного проектирования. Это обусловлено ростом сложности новых программно-технических систем и

высокой ценой ошибок, допускаемых на ранних стадиях их проектирования. По оценкам экспертов, обнаружение и исправление ошибки на этапе проектирования может стоить в 2 раза, на этапе тестирования - в 10 раз и на этапе эксплуатации - в 100 раз дороже, чем на этапе анализа [54].

Для железнодорожного транспорта в настоящее время, в условиях разработки интеллектуальной транспортной системы, актуальным является создание на базе ВУЗов единого научного комплекса для решения прикладных задач и научного сопровождения процессов автоматизации железнодорожного транспорта Украины [136].

2.3 Принципы и общие требования к методологии проектирования систем реального масштаба времени

Рассмотреные изменения в процессах информатизации требуют переосмысления и принципов проектирования информационных систем [111]. Особенно систем реального масштаба времени, к которым относятся АСУ сортировочных станций (см. п. 2.1).

По определению [203] "системой реального времени является такая система, коректность функционирования которой определяется не только правильностью выполнения вычислений, но и тем, в какой момент времени получен необхрдимый результат. Если требования по времени не выполняются, то считается, что произошел отказ системы. Для того, чтобы система могла удовлетворить требованиям, предъявляемым к системам реального времени, аппаратные, программные средства и алгоритмы работы системы должны гарантировать заданные временные параметры реакции системы. Время реакции не обязательно должен быть очень маленьким, но он должен быть гарантированным (что соответствует предъявляемым требованиям) ". То есть основные требования к системам реального времени можно графически отобразить следующим образом (см. рис. 2.4) [187].

Отметим главные принципы проектирования систем:

S - принцип **системного** единства и развития (требует использования открытых моделей и методов проектирования, стандартных протоколов и моделей взаимосвязи открытых систем);

D - принцип **декомпозиции** систем,что проектируются;

I - принцип **иерархичности**;

M - принцип **многоэтапности** процесса проектирования;

К - много критериальность проектирования **с преимуществом временных критериев**;

Р - принцип использования методологии **параллельного** проектирования;

R - принцип **итеративности (рекурсивности)** проектирование;

Т - принцип **типизации** - унификации проектных решений и проектных процедур;

А - принцип **автоматизации** процессов проектирования и формирования информационной базы проектных решений.

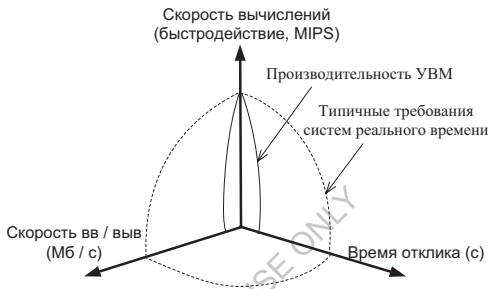


Рис. 2.4 Пространство требований для систем реального времени

Методология проектирования информационных систем должна учитывать изменения,которые происходят. Во-первых, для сложных информационных систем на первый план выходит задача **многоуровневой структурной оптимизации**, которая должна решаться на нескольких уровнях: уровне инфраструктуры объектов автоматизации (топологии размещения объектов), информационной и технической структуры (см. главу 3). Во-вторых, как унифицированная единица объема работы, выполняемой в системе, предлагается использовать «φ-транзакцию» как разветвленную последовательность выполняемых системных и прикладных программ с момента поступления заявки в систему до момента завершения ее обработки, связанной или с занесением информации в базу данных, или с выдачей управляющего воздействия, или отображением информации для персонала системы (см. раздел 4).

Методика проектирования сложных систем должна опираться на **комплекс математических и имитационных моделей**. Имитационное моделирование применяется для решения сложных задач оценки режимов функционирования

распределенных систем. Эти же модели могут затем использоваться для отладки программного обеспечения системы.

Методика должна опираться на средства автоматизации выполнения основных и наиболее трудоемких проектных процедур с формированием базы проектных решений в соответствующей проблемной области.

2.4 Методики концептуалтного проектирования как основа создания сложных ресурсосберегающих АСУ

Современные средства автоматизированного проектирования основаны на различных технологиях, методиках и схемах проектирования, многие из которых доведены до уровня государственных и международных стандартов [194, 206, 165, 164, 209, 196].

Наибольшее распространение при создании сложных систем получил метод функционального структурного анализа и проектирования Д. Росса SADT (Structural Analysis and Design Technique). Основные положения метода SADT, сформулированные почти 20 лет назад, получили развитие в США в рамках проекта ICAM (Integrated Computer-Aided Manufacturing) по созданию интегрированных АСУ на базе компьютерных технологий. В проекте ICAM было разработано семейство методологий IDEF (ICAM DEFinition):

- IDEF0 - методология создания функциональной модели производственной среды или системы (основанная на методе SADT);
- IDEF1 - методология создания информационной модели производственной среды или системы (основанная на реляционной теории Кодда и использовании ER - диаграмм Чена);
- IDEF2 - методология создания динамической модели производственной среды или системы.

Позже в составе проекта ICAM начали разработку проекта IISS (Integrated Information System Support) по созданию технологии логического и физического объединения неоднородных вычислительных систем в вычислительную сеть. В результате этой работы появилась методология семантического моделирования данных IDEF1X как расширение методологии IDEF1.

Инструментальная поддержка методологии IDEF осуществляется с помощью пакета IDEF / Design, разработанного фирмой Meta Software Corporation (США) [78].

Другим направлением развития методологии SADT было создание в Великобритании промышленной информационной технологии SSADM (Structured Systems Analysis and Design Method), которая с 1981 года объявлена открытым отраслевым стандартом, обязательным для использования во всех государственных проектах автоматизированных систем (АС). В 1990 году была принята четвертая версия технологии SSADM, ориентированная на применение инструментальных программных средств проектирования. С середины 1980-х годов работает Ассоциация пользователей SSADM - International SSADM Users Group, которая объединяет более 300 организаций, которые используют SSADM при создании АС. В 1993 году технология SSADM была официально принята в качестве государственного стандарта Великобритании. Эта методология получила признание во многих странах Западной Европы.

Методическое обеспечение технологии SSADM включает тринадцать взаимосвязанных методик проектирования АС: определение требований к АС; моделирование информационных потоков; логическое моделирование данных; определение функций и задач; динамическое моделирование данных; реляционный анализ данных; выбор вариантов автоматизации; разработка демонстрационного прототипа; выбор вариантов технической реализации; проектирование диалогового взаимодействия; логическое проектирование процедур обработки данных; физическое проектирование баз данных; физическое проектирование процедур обработки данных.

Перечисленные методики различаются по степени формализации и возможности автоматизации проектных процедур, которые в них содержатся. Наименее формализованными есть методики, которые относятся к ранним стадиям проектирования систем. Тем не менее, в технологии SSADM все методики имеют четко структурированные проектные процедуры и формализованные критерии оценки качества результатов.

Программные средства автоматизации, что поддерживают различные методики SSADM, разрабатывают несколько десятков фирм в Великобритании, однако, прошли успешную сертификацию только девять программных продуктов. Это говорит о большой наукоемкости и программной сложность всех инструментальных средств автоматизации технологий проектирования систем. Поэтому их стоимость, при небольшом тираже, не превышает ста, достаточно высока и колеблется от нескольких тысяч долларов.

Особенностью американской (IDEF) и европейской (SSADM) технологий проектирования является то, что они, во-первых, охватывают в основном

вопросы проектирования информационного и программного обеспечения АС, и, во-вторых, уделяют большое внимание управлению разработкой и контроля качества проектирования. Вторая особенность проявляется в том, что в технологии SSADM, например, проектная документация разделена на три категории: техническая, организационно-распорядительная и контроля качества. При этом определены конкретные требования к структуре, содержанию и критериев оценки каждого документа. Отдельные технологии проектирования регламентируют только процесс организации корпоративной разработки сложных АС.

На железнодорожном транспорте к таким технологиям можно отнести практическую методику PROMETEO (PROcedure, METhod and Organisation for international computer projects), разработанную международным союзом железных дорог UIC (International Union of Railways) для выполнения проектов информационных вычислительных систем, которые охватывают несколько взаимосвязанных железных дорог. Она основана на следующих методологических принципах [161]: модульность проектов; четкое определение каждой стадии: результаты; данные для следующей стадии; наличие контрольных точек для проверки «Кто? Что? Как?»; структурной связи между различными уровнями контроля проекта; компьютерное приложение методов; "Процедура + Метод + Организация" = практический справочник исполнителя.

Методика PROMETEO предусматривает разбиение проекта на фазы, а каждой фазы - на стадии. Последовательность фаз и стадий разработки проектов по данной технологии представлена в таблице 2.1 и на рисунке 2.2.

Рассмотрены технологии, которые были выбраны только по причинам своей распространенности и комплексности охвата проектно-исследовательских работ, представляют собой лишь наиболее типичную, стандартизированную часть большого количества других моделей, методов и технологий проектирования [153, 59].

В отечественной практике проектирования АС наиболее близкой по своему назначению, содержанию и объему является информационная технология проектирования автоматизированных систем, описана в ГОСТ группы 34 [7]. На рисунке 2.5 представлена схема жизненного цикла сложной АС, описана в терминах трёх технологий: ГОСТ 34, SSADM и PROMETEO.

Рис. 2.5 Стадии проектирования автоматизированных систем

Анализ представленных технологий показывает, что:

а) последовательная (каскадная) схема проектирования [7] не соответствует рекурсивному характеру процесса исследования и разработки сложных систем; современные технологии ориентируются на итерационность [56] и рекурсивность [153] проектирования с параллельным решением проектно-исследовательских задач;

б) технологии проектирования основаны на четкой структуризации процесса создания сложных систем, в котором для каждой стадии (фазы, этапа) проектирования определяется не только ЧТО необходимо получить [7, 163], но и КАК это сделать [56];

Таблица 2.1

Методика PROMETEO

Фаза проектирования		Стадии проектирования	
Обозначение	Наименование	Обозначение	Наименование
P	Подготовка	P1	Представление системы
		P2	Планирование исследований
M	Формирование группы партнеров	M3	Формирование группы партнеров
D	Проектирование	D4	Функциональные исследования
		D5	Архитектурное исследование
I	Реализация	I6	Привлечение средств
		I7	Реализация
		I8	Приемочные испытания и утверждения
		I9	Реализация систем партнеров
		I10	Подготовка к введению в действие
		I11	Введение в действие
O	Эксплуатация	O12	Эксплуатация

в) в вопросе КАК решать проектно-исследовательские задачи при создании сложных систем все подходы декларируют - с использованием ЭВМ, но только IDEF и SSADM из рассмотренных технологий, включают инструментальные средства и подробные методики их применения;

г) наименее формализованными, но наиболее важным с позиций

уменьшения «дорогих ошибок» в проекте, являются ранние этапы анализа и проектирования систем;

д) большинство технологий и инструментальных средств проектирования предназначены для решения только отдельных задач: разработки информационного и программного обеспечения, компоновка технической структуры системы, управление проектами;

е) в качестве моделей в методологиях проектирования систем используются функциональные структурные модели, информационные статические и динамические модели (модели состояний, модели процессов); для их описания широко используются графические схемы и табличные формы [54, 56, 153].

ж) новейшие технологии проектирования (SDLC, E2AF, MDA, RM-ODP, RUP, TOGAF, логическая структура Захмана т.д. [194]) больше внимания уделяют архитектуре систем управления предприятиями и так называемым фреймворка - четким структурным и динамическим моделям многократного использования, в которой системы могут быть описаны и разработаны. То есть фреймворк - это среда, которая содержит математические модели, методы, алгоритмы, инструментальные программы, объединенные в единой методике исследования, проектирование и совершенствование систем.

с) среди известных проанализированных методик отсутствуют методики проектирования сложных АСУ на железнодорожном транспорте, которые работают в реальном масштабе времени.

2.5 Формирование научно-методологического комплекса системного интегратора CoDeCS (**Co**nceptual **De**sign of **C**omplex Real-time Management **S**ystem)

Научно-методический комплекс системного интегратора КСИ - это набор математических моделей, методов, программных инструментальных средств, объединенных в инженерную методику системного анализа, проектирования и совершенствования информационно-управляющих систем реального масштаба времени сортировочных станций.

Эта методика является развитием и обобщением научно-методического подхода к проектированию сложных систем, который разрабатывается на кафедре ЭВМ Днепропетровского национального университета железнодорожного транспорта имени академика В. Лазаряна уже на протяжении

десятков лет. Он нашел применение при разработке таких систем как АСУРСГ (ст. Ясиноватая), АСУМД (ст. Пермь-Сортировочная, Россия), ИУКСС (ст. Нижнеднепровск-Узел), ИПС СС (ст. Орехово-Зуево, Россия), система аналитических серверов АСУ ГП УЗ, корпоративная информационная сеть университета (ДНУЖТ) и других информационно-управляющих систем [59, 108, 60, 115].

В разработанной методике учтены описаные в п. 2.4.2 современные принципы и требования к технологиям создания сложных информационных систем. Эта методика применяется итеративно при выполнении последовательности стадий, которые относятся к системному проектированию и включают (в соответствии с [7]): стадии формирования требований к системе, разработка концепции построения автоматизированной системы, техническое задание, эскизный и технический проект (см. рис. 2.2) .

Критерии оптимизации систем реального масштаба времени разделяют согласно [187] на качественные эксклюзивные критерии (X), качественные критерии поэтапного использования (G) и количественные критерии (Q) (см. рис. 2.6).

Доминирующими критериями и ограничениями в процессе проектирования АСУ реального масштаба времени является временные качественные (X0, G0) и количественные (Q1, Q1.1, Q1.2, Q1.3, Q1.4) характеристики. Они дополняются критериями Q2 (Capacity reserves) по обеспечению необходимых резервов производительности, пропускной способности, емкости (см. рис. 2.4) и, конечно, Q3 - полные проектные стоимости (верхние расходы - "the bottom line").

Стандарту ISO число количественных критериев ограничено двумя метриками: время реакции (продолжительность ответа) и число транзакций, которые нужно обработать в период времени [209].

Рассмотрены критерии и положены в разработанную методику, Макросхема которой представлена на рисунке 2.3.

Концептуальную основу методологии составляют следующие положения.

1. Процесс анализа и проектирования сложных систем представляется в виде последовательно-итерационной схемы поэтапного поиска рациональных проектных решений с использованием эвристических методов оптимизации, которая может быть настроена или связана с последовательностью ранних стадий, этапов, задач информационной технологии проектирования АС по ГОСТ 34. При этом в ранних стадий отнесены стадии 1, 2, 3, 4, 5 (см.рис.2.5).

Рис. 2.6 Критерии оптимизации систем реального времени

2. База знаний методологии представляется в виде набора онтологий сортировочных станций, сортировочных горок и комплекса программно-технических средств промышленных компьютеров Advantech [158, 84], на которые ориентированы разработки систем автоматизации сортировочных станций в Германии, России и Украины.

3. Методология является адаптивной к набору исходных данных, имеющихся у разработчика при анализе и проектировании конкретной системы. Это означает, что в базе знаний активизируется и последовательность проектных этапов и процедур, для которых есть все необходимые данные (или в базе знаний, или введены проектировщиком в режиме диалога).

4. Реализация методологии представляется в виде программы CSI комплекса системного интегратора, программы CSProject для построения и расчета информационно-временных характеристик φ -транзакций компьютерных систем реального масштаба времени, программы OPTiFLOW распределения информационных потоков в системах реального времени, программы ОПТИКОС оптимизации информационно-управляющих систем, программы GAOSIS генетического алгоритма оптимизации структур

информационных систем и программы PRIORY выбора приоритетов потоков заявок в информационных системах.

5. Исходные данные и результаты системного анализа и проектирования выдаются в таблично-графических формах.

Исходными данными для создания системы являются:

блок 1 - характеристика объекта автоматизации: описание организационно-технологических характеристик системы, что проектируется, включающий технологические подсистемы, их структуры, участки, потоки и характеристики сигналов на участках, средства механизации и оборудование низовой автоматики, топология их размещения, помещение на станции , показатели надежности устройств, последовательность технологических операций, выполняет оперативный персонал системы, которая проектируется (потоки, место, время); периферийное оборудование, которое должно обслуживаться АСУ;

блок 2 - характеристика функционирования АСУ (с позиций заказчика): функции, задачи, алгоритмы или программы, которые должена выполнять система, что проектируется; для новых систем это описание будет на уровне функций и задач; для систем, которые совершенствуются или модернизируются - это разработанные алгоритмы или программы по оценке частотного состава операций (команд); для всех уровней с разной степенью полноты и точности задаются компоненты информационного обеспечения (массивы); для каждого сигнала (заявки) указываются управляющие действия или сообщение персоналу, что выдаются; описание оформляется в виде φ -транзакций в диалоге с программой CSI;

блок 3 - описание технических средств автоматизации: управляющие ЭВМ, микроконтроллеры, средства ввода-вывода, нормализации и коммутации сигналов, устройства питания, конструктивы, сетевое оборудование, программное обеспечение, показатели надежности и стоимости устройств;

блоки 4, 5, 6, 7 - описывают критерии и ограничения в бизнесе (Q2, Q3 - показатели ресурсосбережения, расходы, потери, стоимость системы, показатели надежности, которые необходимо обеспечить), технологические (элементы топологии, каналы), функциональные (Q1, Q2) и технические (загруженность процессоров, каналов, стоимость программно-технических средств, унификация решений);

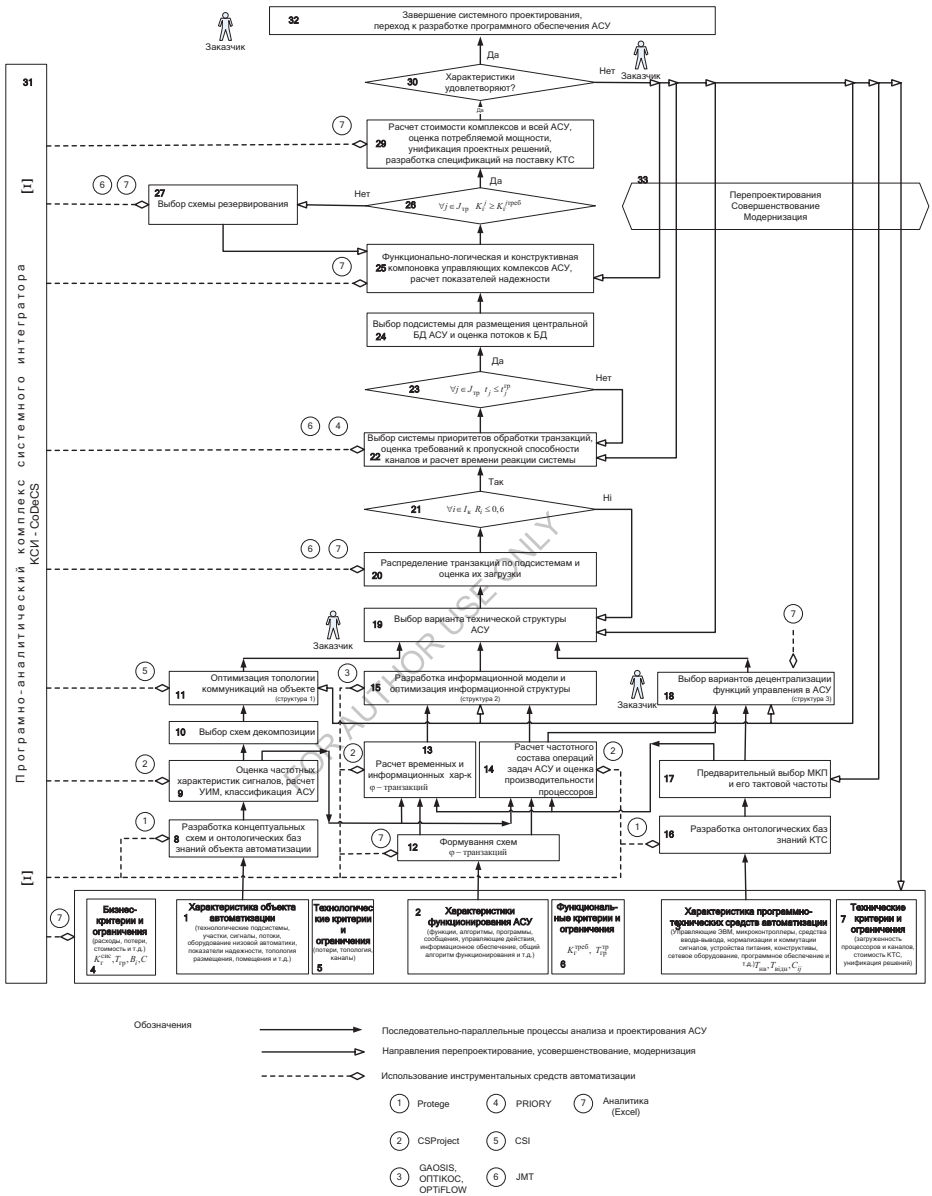


Рис. 2.7 Научно-методический комплекс системного интегратора CoDeCS

Проектирование системы начинается с тремя взаимосвязанными направлениями. Первый - блоки 8, 9, 10, 11: оптимизация структуры коммуникаций на объекте автоматизации по критерию минимизации суммарной длины коммуникаций с фиксированными каналами (построение минимального скелетного дерева). Второй - блоки 12, 13, 14, 15: оптимизация информационной структуры АСУ по критерию минимизации прироста суммарного информационного потока при уменьшении количества информационных связей в структуре (см. п. 3.3, 3.4). Третье направление - блоки 16,17,18: выбор вариантов децентрализации функций в иерархической структуре по критериям минимизации суммарных потерь или стоимости систем (см. п. 3.5).

В рамках первого и третьего направлений формируются в соответствующие онтологии сортировочной станции и горки (см. П. 2.3) и комплекса программно-технических средств [84].

Первая онтология используется в блоках 9, 10, 11, 19, 24, 25, 29.

Вторая онтология - в блоках 17, 18, 24, 25, 29.

На основе сформированных трех вариантов структуры избирается с участием заказчика, который представляет текущие бизнес-интересы (приоритеты), вариант технической структуры (блок 19) для предварительно выбранного типа процессора (блок 17).

На втором направлении на основе исходных данных (блок 2) формируются φ-транзакции (блок 12), описанные в разделе 4, и рассчитываются их временные и информационные характеристики (блок 13) для оптимизации информационной структуры (блок 15). Кроме того, рассчитываются частотный состав операций, используемых в системе, и производительность процессоров в метриках MIPS и в φ-транзакциях / с для поиска оптимальных иерархических структур (блок 18).

Для выбранного варианта технической структуры φ-транзакции распределяются по подсистемам и оценивается их загрузка (блоки 20, 21). Если это ограничение выполняется, для распределенной АСУ для каждой φ-транзакции рассчитывается время реакции системы, выбирается оптимальный приоритет и вычитаются требования к пропускной способности каналов (блок 22). Далее, при выполнении временных ограничений (блок 23), на основе данных блока 13 выбирается подсистема для размещения центральной базы данных в децентрализованной АСУ (блок 24).

На следующем этапе (блок 25) для каждой подсистемы выполняется

функционально-логическая и конструктивная компоновка соответствующего компьютерного комплекса с использованием онтологии КПСА (блок 16). Кроме этого избирается технология и средства связи в системе с той же онтологией. Эти средства должны соответствовать требованиям по пропускной способности каналов (блок 22). Для полученной технической структуры выполняется оценка коэффициентов готовности и, если они не обеспечивается (блок 26), выбирается схема резервирования слабого элемента (блок 27).

Если нужна надежность обеспеченная, то для АСУ выполняется унификация проектных программно-технических решений и составляются спецификации, рассчитываются стоимости комплексов и всей системы, потребляемая мощность и эксплуатационные показатели системы (блок 29). Результаты проектирования рассматривает заказчик (блок 30) и, если они его устраивают, системное проектирование завершается и начинается разработка программного обеспечения (блок 32). Если "Нет", выбирается направление перепроектирования (блок 32). С этого же места начинается и совершенствования или модернизация системы.

На рисунке 2.7 указаны инструментальные средства, которые используются в процессе исследования и проектирования АСУ.

Разработанный научно-методический комплекс системного интегратора (КСИ) используется не только для системного проектирования и совершенствования систем автоматизации сортировочных станций, но и в учебном процессе при подготовке специалистов специальности «Компьютерные системы и сети» направления «Компьютерная инженерия» в Днепровском национальном университете железнодорожного транспорта имени академика В. Лазаряна при преподавании дисциплин «Проектирование информационно-управляющих комплексов», а также в курсовом и дипломном проектировании.

3. Методы построения и исследования информационно-функциональных и технических структур АСУ

3.1 Основные принципы построения М-моделей распределенных информационно-управляющих систем

В процессе проектирования в соответствии с разработанной методикой (рис. 2.7), заказчик должен подготовить схемы информационных потоков в существующей и будущей системе без привязки к используемым техническим средствам, подлежащих замене в новой системе. То есть для заказчика и проектировщика мы имеем дело с несуществующей пока системой, которую назовем мета системой. Тогда информационную модель потоков в ней будем называть мета модели или М-моделью.

На рис. 3.1 показана информационная мета система (ИЦС), который взаимодействует с технологическим объектом автоматизации (ТОА). ИЦС описывается информационной мета моделью. В мета модели объект автоматизации ТОА отображается в виде двух не пустых множеств - источников и приемников информации $I = \{i_1, i_2, ..., i_n\}$, $\Pi = \{\pi_1, \pi_2, ..., \pi_m\}$.

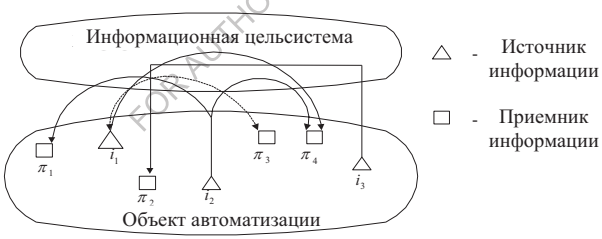


Рис. 3.1 Метамодель системы, что проектируется

Каждый источник i_k может генерировать множество ориентированных (направленных) информационных потоков $P_k = \{p_{k1}, p_{k2}, ..., p_{kl}\}$. Каждый информационный поток описывается четверкой:

$$\langle OD_{kj},\ b_{kj},\ \lambda_{kj},\ X_{kj} \rangle, \qquad (3.1)$$

где - $OD_{kj} = (i_k, \pi_g)$ - это OD-пара или Origin-Destination пара, указывающая источник i_k и π_g приемник информационного потока;

b_{kj} - средний объем сообщения (информации), генерируемого источником i_k, который создает поток p_{kj} (бит); каждый источник может генерировать несколько потоков, отличаются в следующих элементах: π_g (назначением), b_{kj} (объемом), λ_{kj} (интенсивностью), X_{kj} (характеристиками);

λ_{kj} - средняя интенсивность генерации сообщений в потоке p_{kj} (1/с);

$X_{kj} = \{W_{kj}^P, T_{kj}^{\text{гр}}, K_{kj}^{\text{i}}, K_{kj}^{\text{г}}\}$ - вектор дополнительных характеристик потока p_{kj} : W_{kj}^P - безразмерный коэффициент, характеризующий "вес" или "важность потока", $T_{kj}^{\text{гр}}$ - предельное время задержки обработки и передачи сообщения b_{kj}, K_{kj}^{i} - коэффициент допустимого искажения (потерь) информации в сообщении при его передаче / обработке $(0 \le K_{kj}^{\text{i}} < 1)$, $K_{kj}^{\text{г}}$ - допустимый коэффициент готовности мета системы при передаче / обработке потока ($0 < K_{kj}^{\text{г}} \le 1$).

X_{kj} включает характеристики, которые требуют дополнительных исследований ТОА и работы с экспертами, и которые не всегда удается получить при проектировании реальных систем.

Очевидно (см. рис. 3.1), что каждый источник может генерировать несколько потоков, предназначенных для одного или нескольких приемников. Приемники же могут принимать несколько потоков от одного или нескольких источников. Предполагать, что в ЦУС процесс обработки не влияет на динамику потоков (см. рис. 3.2). То есть, мета ресурсы (гипотетические ресурсы обработки данных), достаточные для реализации любых функций обработки.

Рис. 3.2 Мета-ресурсы

На данном информационном уровне не решаются задачи, где будут находиться устройства обработки данных (в точке i_k или в точке π_g) и какую

они должны иметь вычислительную мощность. Они рассматриваются на других этапах методики (рис. 2.7) [80].

Таким образом, начальную мета модель системы можно представить в виде ориентированного несвязного мультиграфа с помеченными ребрами (см. рис. 3.3а). Поскольку на реальных ТОА источники и приемники могут находиться территориально в одном месте (комнате, кроссовом шкафу и т.п.), то в начальном мультиграфе сведем все такие точки до одного элемента, который будем называть **агрегированным узлом** (см. рис. 3.3б, рис . 3.3в). Полученная в этом случае мета модель будет состоять из множества узлов. Уточним: **узел** - технологическая подсистема (ТПС), в которой находится **источник** и/или **приемник** и / или которых дополнительно может быть **транзитом**. Последний тип узла может появиться в процессе структурной оптимизации. Он отвечает подсистеме, которая является не только источником и / или приемником, но и промежуточным узлом в передаче информационных потоков. Таким образом множество узлов (вершин) мета модели (графа) можно описать как

$$U = \{u_1, u_2, ...u_{N_U}\} = I \cup \Pi \cup U_a \setminus I_a \cup \Pi_a, \qquad (3.2)$$

где U_a - множество вершин, полученных в результате агрегации;

I_a, Π_a - множество поглощенных (агрегированных) источников и / или приемников.

Для анализа и синтеза информационной структуры и оптимального перераспределения информационных потоков, введем два новых понятия: **линк** и **роут**.

Линк - это двунаправленный (дуплекс) информационный мета канал, соединяющий два узла, между которыми передаются информационные потоки. Фактически наличие линка в модели определяется тем, существует ли информационный поток между его узлами. В мета модели множество линков можно задать таким образом:

$$L^0 = \{l_j\},\ j = \overline{1, N_L},\ N_L = \left|L^0\right|, \qquad (3.3)$$

$$l_j = \{(u_{j1}, u_{j2}), P_j, Y_j\}, \qquad (3.4)$$

где (u_{j1}, u_{j2}) - узлы (вершины), которые соединяет мета канал или линк;

$P_j = \{\overline{p}_{j1}, \overline{p}_{j2}, ..., \overline{p}_{jk}\}$ - множество потоков роута, проходящих по *j*-му линку в двух направлениях;

$Y_j = \{\vec{C}_j^{\text{тр}}, \overleftarrow{C}_j^{\text{тр}}, K_j^{\text{пом}}, K_j^{\text{г}}, W_j^L\}$ - множество дополнительных характеристик мета канала, отражающие особенности его возможной технической реализации;

$\overleftarrow{C}_j^{\text{тр}}, \vec{C}_j^{\text{тр}}$ - предельные значения пропускной способности линка в двух направлениях; очевидно, что должны выполняться ограничения:

$$\overleftarrow{C}_j^{\text{тр}} \geq \sum_{p_{ji} \in P_j} \overleftarrow{p}_{ji}, \tag{3.5}$$

$$\vec{C}_j^{\text{тр}} \geq \sum_{p_{ji} \in P_j} \vec{p}_{ji}, \tag{3.6}$$

$K_j^{\text{пом}}$ - коэффициент появления ошибок при передаче сообщений по мета канала (ош/ бит);

$K_j^{\text{г}}$ - коэффициент готовности канала ($0 < K_j^{\text{г}} \leq 1$);

W_j^L - весовой коэффициент ("вес", "стоимость") мета канала.

Вектор характеристик Y_j, как и X_{kj}, требует дополнительных исследований ТОА и знания особенностей технических средств для построения управляющего вычислительного комплекса (УВК). Поэтому в полном объеме Y_j могут применяться при итерационном проектировании систем по разработанной методике.

Роут - это информационный поток, описывается четверкой (3.1) и дополнен множеством транзитов. В мета системе функционирует множество роутов:

$$R^0 = \{R_1, R_2, ..., R_{N_R}\}, \tag{3.7}$$

где, согласно (3.1),

$$R_i = \{O_i, D_i, b_i, \lambda_i, X_i, T_i\} = \{OD_i, \overline{P}_i, T_i^{(u)}, X_i\}; \tag{3.8}$$

$\overline{P}_i = b_i \times \lambda_i$ - информационный поток (бит/с);

$T_i^{(u)} = \{u_{i1}, u_{i2}, ..., u_{iN_T}\}$ - множество транзитных узлов (не включают OD-пару), по которым передается информационный поток; в начальной модели $T_i^{(u)} = \varnothing$; вместо $T_i^{(u)}$ можно использовать $T_i^{(l)} = \{l_{i1}, l_{i2}, ..., l_{iN_T+1}\}$ - множество транзитных линков, тогда в начальной модели $|T_i^{(l)}| = 1$.

Очевидно, что $N_R \geq N_L$.

Таким образом полная начальная мета модель информационной системы с учетом (3.2) - (3.8) может быть представлена в виде неориентированного реберно-взвешенного графа с ориентированными роутами (см. Рис. 3.3В)

$$G = (U, L^0, R^0), \tag{3.9}$$

Следует отметить, что начальный граф может быть несвязанным, что говорит о наличии в ЦУС автономных подсистем. В этом случае структурная оптимизация или выполняется в пределах одной подсистемы, или в задании должны вводиться дополнительные условия появления новых линков и построения связного графа (см. рис. 3.3г). В дальнейшем будем предполагать, что граф мета модели - связный.

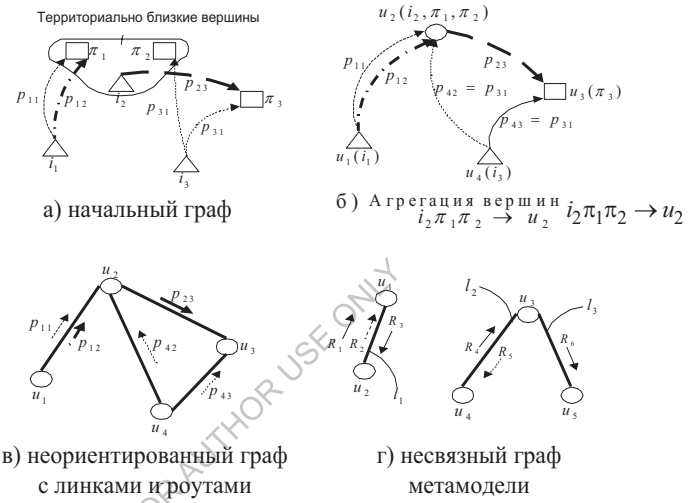


Рис. 3.3 Преобразование М-модели

3.2 Задача структурной оптимизации М-модели и общий алгоритм ее решения

В работе [81] поставлена и решена задача синтеза структуры системы на начальном графе (3.9) как поиск такого минимально связного графа $G^* = (U, L^*, R^*)$, у которого:

$$\{\sum_{R_i^* \in R^*} p_i \times (\left|T_i^{(u)*}\right| + 1) + \sum_{R_j \in R^*} p_j\} - \sum_{R_i \in R^0} p_i \rightarrow \min, \qquad (3.10)$$

$$|L^*| \rightarrow \min. \qquad (3.11)$$

При последующих "ослабленных" ограничениях:

- $L^* \subset L^0$ - поиск проводится на начальной множестве линков; линки могут удаляться, но не добавляться, в графе;

- $R^* = \{R_i^*\}_{i=\overline{1,N_i}} \cup \{R_j\}_{j=\overline{1,N_j}}$ и $N_i + N_j = N_R$ роуты не расщепляются; тут R_i^* – роут, у которого $T_i^{(u)*} \neq \varnothing$ или $T_i^{(l)*} > 1$;

- роуты не образуют циклов $N_{T_i^{(u)*}} \leq N_U - 2, \quad \forall i = \overline{1, N_i}$.

Фактически задача состоит в поиске такого минимально связного графа путем удаления ссылок и перераспределении потоков, в котором увеличение суммарного потока в графе будет минимальным.

Сформулирован критерий может иметь инвариантные формы записи в зависимости от способа описания транзитов ($T_i^{(u)*}$ или $T_i^{(l)*}$). Если в (3.10) перейти от $T_i^{(u)*}$ к $T_i^{(l)*}$, то эквивалентный критерий оптимизации будет выглядеть:

$$\sum_{i=1}^{N_R} p_i \times \left|T_i^{(l)*}\right| - \sum_{i=1}^{N_R} p_i \to \min.$$

Вынося за скобки общий множитель, получим:

$$\sum_{i=1}^{N_R} p_i \times (\left|T_i^{(l)*}\right| - 1) \to \min. \qquad (3.12)$$

В этом случае вычисления ведутся по всем роутам без разбивки на R_j и R_i^* (нераспределенные и распределены роуты), что несколько упрощает алгоритм поиска оптимальной структуры.

Критерии (3.10) и (3.12) предполагают использование параметров роутов p_i и $T_i^{(u)*}$, $T_i^{(l)*}$. Однако критерий имеет еще одну эквивалентную форму записи через характеристики линков, описание которых (3.4) включает и направленные потоки, проходящие по ним. Тогда минимизацию прироста суммарного потока в графе можно записать так

$$\sum_{l_j \in L^*} C_j^* - \sum_{l_j \in L^0} C_j \to \min, \qquad (3.13)$$

$$\sum_{l_j \in L^*} \Delta C_j \to \min. \qquad (3.14)$$

З (3.4)-(3.6) видим, что $C_j = \vec{C}_j + \overleftarrow{C}_j = \sum_{i=1}^{k} p_{ji}$. При появлении в линке новых потоков, P_j дополняется новыми элементами и k растет.

Таким образом, в данной постановке задачи с "слабыми" ограничениями есть возможность выбора критерия оптимизации из полученных вариантов

(3.10), (3.12) или (3.14) с учетом особенностей построения структур данных и алгоритмов поиска решений.

Рассмотрим один из алгоритмов решения задачи, основанный на имитации удаления наименее загруженного линка ($C_j \to \min$) без нарушения связности графа и с перераспределением его роута по кратчайшим путям. Обобщенный алгоритм решения задачи можно представить в виде следующей последовательности.

1. Сформировать $L^{(\text{вил})}$ (выбрать линки, изъятие которых не нарушает связности)

2. Пока $L^{(\text{вил})} \neq \varnothing$ выполнять:

2.1 Упорядочить линки по C_j

2.2 $\forall l_i \in L^{(\text{вил})}$

2.2.1 Имитация изъятия l_i (с поиском l_{opt} с минимальным приростом (3.14))

2.2.2 $\forall p_j \in P_i$ $(\forall R_j \in R_i)$

2.2.2.1 Поиск кратчайшего маршрута (с учетом T_j)

2.2.2.2 Перераспределение роутов по новым маршрутам

2.2.3 Вычисление $\sum_{l_q \in L^*} \triangle C_q$

2.2.4 Восстановление l_i и всех $R_j \in R_i$

2.3 Определение l_{opt}, у которого $\sum_{l_q \in L^*} \triangle C_q \to \min$

2.4 Изъятие l_{opt}, перераспределение его роутов по кратчайшим маршрутам, формирование T_i^*

2.5 Вычисления общего критерия относительно начального состояния.

2.6 Сформировать $L^{(\text{вил})}$

3. Конец.

3.3 Методы поиска рациональных вариантов информационных структур АСУ на основе генетических алгоритмов

3.3.1 Постановка задачи и метод рационального распределения информационных потоков в информационно-управляющих системах

Как уже отмечалось в п. 3.2, исследуемый объект оптимизации в исходном состоянии представляется в виде множества узлов типа источника и приемники, и множества безтранзитних роутов. Поскольку некоторые узлы могут быть одновременно источниками и приемниками, или один источник может иметь несколько приемников, исходную структуру можно представить в виде ориентированного реберно-взвешенного мультиграфа. В данной части рассмотрим генетический подход к поиску рациональных вариантов информационных структур, отличающийся использованием интегрального критерия оптимизации, которые совмещают задачи блоков 11 и 15 (рис. 2.7) и формируют оптимальный вариант с учетом суммарной длины каналов и прироста суммарных информационных потоков, которые в них передаются [115].

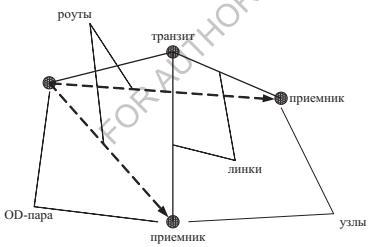


Рис. 3.4 Начальный граф для оптимизации

Введем обозначения с уточнениями:

$G = (U, L^0, R^0)$- выходной ориентированный реберно-взвешенный мультиграф;

$U = \{u_1, u_2, ..., u_{n_U}\}$- множество узлов;

$L^0 = \{l_j\}, j = 1,2,..., N_L$ - множество линков,

где $l_j = \{(u_{j_1}, u_{j_2}), c_j, len\}, a\ c_j = \vec{c}_j + \overleftarrow{c}_j$;

$R^0 = \{r_1, r_2, ..., r_{N_R}\}$- множество роутов, где

$r_i = \{o_i, d_i, p_i, T_i\}$- i-й роут, в котором o_i,d_i - пара OD- узлов или OD-пара (Origin – Destination), а p_i – потік информации между узлами OD-пары;

$T_i = \{u_{i_1}, u_{i_2}, ..., u_{N_T}\}$- множество транзитов.

На основе введенных определений и обозначений задача оптимизации распределения информационных потоков в системе, автоматизируется, формулируется следующим образом.

Для заданного ориентированного реберно-взвешенного мультиграфом $G = (U, L^0, R^0)$, у которого $\forall i = 1,2,...,N_R$, $T_i = \varnothing$ та $N_R \geq N_L$, найти такой минимально связный граф $G^* = (U, L^*, R^*)$, чтобы

$$Pk = \left[\sum_{r_i^* \in R^*} p_i \times \left(\left| T_i^* \right| + 1 \right) + \sum_{r_j^* \in R^*} p_j \right] - \sum_{r_i \in R^0} p_i \to \min \qquad (3.16)$$

При этом заданы следующие ограничения:

$R^* = \{r_i^*\}_{i=\overline{1,N_i}}$ та $N_i + N_j = N_R$ (роуты не розчиплюються),

где $r_i^* = \{o_i, d_i, p_i, T_i^*\}$, $T_i^* \neq \varnothing$ (потоки не меняются по величине), причем $N_{T_i^*} \leq N_U - 2$, $\forall i = \overline{1, N_i}$ (роуты не создают циклов).

Также можно ввести критерий минимизации по суммарной длине линков:

$$Lk = \sum_{len_i^* \subset L^*} len_i \to \min . \qquad (3.17)$$

Окончательный критерий для определения оптимальности решения выглядит следующим образом:

$$K = (C_1 \cdot Pk + C_2 \cdot Lk) \to \min . \qquad (3.18)$$

Описанный набор ограничений будем называть "ослабленными ограничениями", поскольку они не учитывают возможные технические ограничения на значения c_j (или отдельные составляющие).

Предложенный критерий (3.16) синтеза структуры, которая содержит минимальное количество линков как можно более короткой длины, которые обеспечивают передачу всех роутов и обладают минимальным суммарным приростом информационных потоков, обеспечивает получение и более высоких показателей надежности реализации роута. Это связано с тем, что выражение (3.16) обеспечивает минимальную длину цепочек прохождения роутов.

Алгоритм решения задачи оптимального распределения информационных потоков основан на имитации удаления ссылок и перераспределении роута с откатом и оценке суммарного прироста информационных потоков для каждого

линка - кандидата на удаление. Для сокращения времени поиска оптимального варианта используются процедуры упорядочения и направленного перебора.

В данном методе, как базовый, используется генетический алгоритм. Кроме этого при решении используется несколько "классических" алгоритмов по теории графов: проверка графа на связность и нахождения маршрутов прохождения роута.

Данный алгоритм отличается от предложенного в п. 3.2 [80], прежде всего, тем, что ограничения на поиск решений только на множестве L^0 не налагается. То есть в структурном графе системы могут присутствовать ссылки, не входящих в заданной условием первичной множества ссылок L^0.

Проектирование хромосомы. Как известно, проектирование хромосомы - сложное и ответственное дело, ведь от ее построения зависит скорость поиска решения.

В данной задачи хромосома кодирует решение в виде квадратной матрицы смежности, которая развернутая в одну ленту. Матрица смежности A (рис. 3.5) представляет собой квадратную матрицу n x n, состоящий из "0" и "1", где n - количество вершин графа (узлов информационно-управляющей системы - ИКОС). Элемент a_{ij} указывает на наличие дуги (линка) между вершинами i и j.

$$A = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n-1} & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n-1} & a_{2n} \\ \dots & \dots & \dots & \dots & \dots \\ a_{n-1,1} & a_{n-1,2} & \dots & a_{n-1,n-1} & a_{n-1,n} \\ a_{n1} & a_{n2} & \dots & a_{n,n-1} & a_{nn} \end{bmatrix} \quad a_{ij} = \begin{cases} 1, \text{ если линк есть } l_{ij} \\ 0, \text{ если нет линка } l_{ij} \end{cases}$$

Рис. 3.5 Матрица смежности

Таким образом хромосома выглядела бы так (рисунок 3.6).

a_{11}	a_{12}	...	a_{1n}	a_{21}	a_{22}	...	a_{2n}	...	a_{n1}	a_{n2}	...	a_{nn}

Рис. 3.6 Вид не оптимизированной хромосомы

Длина хромосомы составит n^2. Но в таком виде хромосома не оптимальна по длине. При указанных условиях задачи матрица смежности является симметричной относительно главной диагонали, а элементы самой главной диагонали равны нулю. Поэтому в хромосоме присутствует много лишней

(элементы главной диагонали) и избыточной (элементы матрицы, находящихся под главной диагональю) информации. Эти элементы на рисунке 3.9 обозначены серым цветом. Для сокращения длины хромосомы решено считать лишь элементы матрицы, находящихся над главной диагональю. На рисунке 3.9 эти элементы выделены жирным шрифтом. Исходя из этого, хромосома будет выглядеть следующим образом (рисунок 3.7).

a_{12}	a_{13}	…	a_{1n-1}	a_{1n}	a_{23}	…	a_{2n-1}	a_{2n}	$a_{i,i+1}$	…	a_{i-1n}	a_{in}	$a_{n-1,n}$

Рис. 3.7 Оптимизированная хромосома

Длина оптимизированной хромосомы составляет:

$$Lc = \sum_{i=1}^{n-1}(n-i) = \frac{n(n-1)}{2}. \qquad (3.19)$$

Это значение бесспорно меньше чем длина не оптимизированной хромосомы. С такой хромосомы достаточно легко восстановить матрицу смежности, перебирая хромосому слева направо. Сначала выделяем (n-1) битов и формируем первую строчку, далее - (n-2) битов и формируем вторую. Так продолжаем до тех пор, пока не разберем всю хромосому.

Таким же образом можно получить множество линков *L*. Получив матрицу смежности или множество ссылок, определяются маршруты следования роута, информационные потоки каждого линка и др.

Разработка фитнес-функции. От построения хромосомы, то есть кодирования развязки, зависит и работа фитнес-функции. Фитнес-функция - единственная часть программы, которая знает "что же происходит на самом деле". Все операторы генетического алгоритма оперируют просто двоичной лентой, а фитнес-функция фактически оценивает качество развязки, для того, чтобы предоставить особи, представленной хромосомой, определенные шансы на выживание и продолжение эволюции.

В данной задачи фитнес-функция построена по алгоритму, изображенным на рисунке 3.8. Критерии *Pkk* и *Lkk*, что приведены в алгоритме, вычисляются немного по другому принципу.

Критерий *Pkk* вычисляется по формуле (3.20).

$$Pkk = \sum_{p_i \in R^0} p_i \Big/ Pk \quad . \qquad (3.20)$$

Таким образом, получаем критерий *Pkk*<1, что является удобным для

представления значения фитнес-функции. Чем ближе значение приспособленности к 1, тем "качественнее" решение.

Расчет *Lkk* построено на том же принципе приведения критерия к диапазону [0,1] (3.6).

$$Lkk = 1 - \frac{\sum_{len_i \in L^*} len_i}{\sum_{len_j \in L^0} len_j}. \qquad (3.21)$$

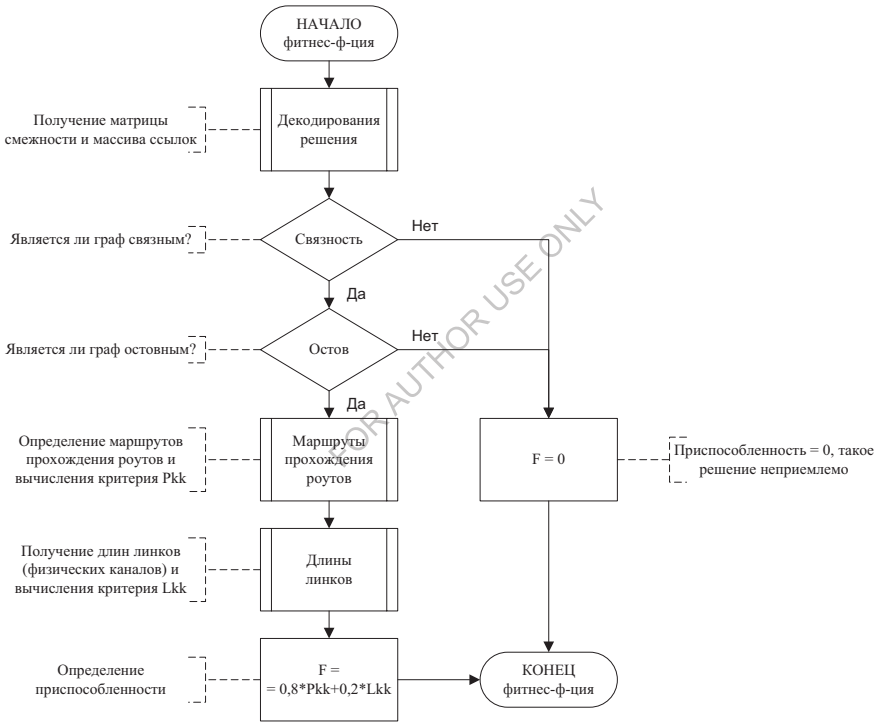


Рис. 3.8 Алгоритм вычисления приспособленности

Окончательное значение фитнес-функции F (3.22) также находится в диапазона [0,1]. Можно регулировать важность критериев путем изменения значений коэффициентов при *Pkk* и *Lkk*.

$$F = C_1 \cdot Pkk + C_2 \cdot Lkk\,. \qquad (3.22)$$

Следует подробнее остановиться на решениях, которые не удовлетворяют условию задачи. Приспособленность этих решений равна 0. К таким решениям относятся несвязные графы и графы, ребра которых образуют циклы. Несвязные графы - это такие, в которых есть хотя бы одна вершина, с которой недоступны все другие вершины. Граф, содержащий n вершин и (n-1) ребер (то есть не содержит циклов), называется деревом или костяком.

Проверка графа на связность происходит путем поиска длин путей между всеми вершинами с помощью алгоритма Флойда-Уоршела [31]. Для этого матрица смежности превращается в матрицу расстояний, где все "1" заменяются на расстояние между соответствующими вершинами, а "0" - каким относительно большим в расстояний числом (например, 10^9). Далее вычисляются длины путей между всеми вершинами с помощью простой процедуры:

```
for k:=1 to NodeCount do
for i:=1 to NodeCount do
for j:=1 to NodeCount do
D[i,j]:=min(D[i,j],D[i,k]+D[k,j]);
```


Если, просматривая всю матрицу расстояний D, найдено хотя бы один элемент, равный 10^9, то пути между соответствующими вершинами нет, а граф является несвязным.

Проверка, является ли граф скелетным, выполняется просто: для этого у связного графа должно быть ровно (n-1) ребер.

Хотя и нецелесообразно использовать генетический алгоритм на задачах, для которых функция приспособленности сложно или долго исчисляется все же решено применить эволюционный подход с целью изучения возможностей генетических алгоритмов при решении подобного класса задач.

Выбор типов операторов генетического алгоритма. От выбора тех или иных типов основных операторов генетического алгоритма зависит, как скорость поиска решения, так и качество развязки. С множества вариантов генетических алгоритмов за основу выбрана канонический генетический алгоритм (фиксированный размер популяции, фиксированная разрядность генов, особи для скрещивания выбираются случайным образом, одно точечный кроссовер и одно точечная мутация) с некоторыми модификациями. Все операторы генетического алгоритма решено писать самостоятельно с целью более глубокого понимания механизмов работы базовых манипуляций над

особями.

Начальная популяция задается путем создания определенного числа особей с применением генератора случайных чисел и вычисления приспособленности особей:

```
SetLength(Population,PopSize);          // численность популяции
for i:=0 to PopSize-1 do                // для каждой особи
with Population[i] do begin             // в популяции
Chromosome:=GeneratePopItem(0.25);      //определить ее генотип
Fittness:=ItemFittness(Chromosome);     // и приспособленность
end;
```

Хромосомы особей генерируются с учетом вероятности равенства биту "1". В общем случае вероятность единичного бита равна 0,5. Но, учитывая условия задачи, в частности то, что граф должен быть скелетным (то есть количество ребер должна быть малой), решено генерировать хромосомы с вероятностью единичного бита не более 0,25 для возможного появления уже в начальном поколении особей с ненулевой приспособленностью. Такое решение способствует быстрому поиску решения.

Исходя из того, что в поколении может быть много особей с нулевой приспособленностью, отбор для скрещивания происходит случайным образом. Каждая особь, независимо от приспособленности, имеет одинаковые шансы на образование потомков. Ведь даже "неприспособленные" особи могут при скрещивании дать "качественный" решение.

Скрещивание особей происходит путем равно точечного кроссинговера. Для этого случайным образом выбирается точка кроссинговера, что разделяет хромосому на две части. Родители обмениваются одними из частей, образуя два потомка.

После скрещивания с некоторой вероятностью может произойти мутация. Обычно, вероятность мутации находится в пределах от 0,01 до 0,1. Вероятность мутации возрастает, если родительские особи похожи. Сходство родителей определяется величиной хемингового расстояния между хромосомами. Если хемингово расстояние не превышает 5% от длины хромосомы, то вероятность мутации растет в несколько раз (в данной реализации - в 5 раз). Такой шаг позволяет средней приспособленности популяции расти быстрее и предупреждает явление сходимости, когда все особи в поколении имеют

одинаковые хромосомы.

Операции отбора и скрещивания происходят (*PopSize* **div** 2) раз, где *PopSize* - размер популяции. Таким образом, образуются столько же потомков, сколько и родителей.

Формирование следующего поколения происходит по следующей схеме. Сначала массивы родителей и потомков упорядочиваются по убыванию значения приспособленности. Далее, к новому поколению отбираются половина родителей и половина потомков. Итак, к новому поколению попадут лучшие особи: как родители, так и потомки.

Выбор критерия остановки и настройка параметровгенетического алгоритма. Критерий остановки работы генетического алгоритма определяет момент окончания развития популяции. В качестве критериев остановки, возможных в программной реализации задачи, приняты следующие:

- достижение максимальной приспособленностью определенного значения, которое задает пользователь;

- прохождение определенного количества поколений (значение задает пользователь);

- комбинация двух предыдущих критериев: если сработает хоть один из них, то процесс поиска решения завершается.

На эффективность работы генетического алгоритма влияют следующие его параметры, подвергающиеся настройке:

- размер популяции (10 - 50);

- вероятность кроссинговера (0,5 - 0,9);

- вероятность мутации (0,01 - 0,1).

Работа по настройке параметров возлагается на пользователя, именно он задает показатели работы алгоритма, в зависимости от желания.

Анализ генетического алгоритма программы ОПТИКОС. Некоторые замечания по эффективности разработанного генетического алгоритма для решения задачи об оптимальном распределении информационных потоков.

В ходе проведения экспериментов по проверке эффективности генетического алгоритма, используемого программой "ОПТИКОС", выявлено интересную закономерность. На скорость нахождения решения влияют не только параметры алгоритма, но и способ генерации начального поколения. Если быть более точным, то именно вероятность единичного бита. Очевидно, что при примененной схеме кодирования решений, для получения скелетной дерева единичных битов (тех, что уровни "1") должно быть меньше, чем нулевых. При

этом приемлемый решение будет достигаться быстрее. Так как количество ребер в скелетной дереве равно (n-1), то рекомендуется вероятность единичного бита вычисляется по формуле:

$$p_1 = \frac{n-1}{Lc} = \frac{n-1}{\frac{n\cdot(n-1)}{2}} = \frac{2}{n}$$, где n – количество вершин (узлов) графа

В качестве базовой задачи для исследования зависимости скорости развязки от вероятности единичного бита и всех опытов, приведенных ниже, выбрано задачу с 10 узлами и 17 роутами. Граф задачи изображен на рисунке 3.9. Опыт выполнялся на компьютере Intel Celeron 433 MHz с 192 MB оперативной памяти под операционной системой Windows XP Professional.

Так, при использовании рекомендованной вероятности генетический алгоритм гораздо быстрее находил ненулевой решение, значительно ускоряло дальнейшую эволюцию популяции.

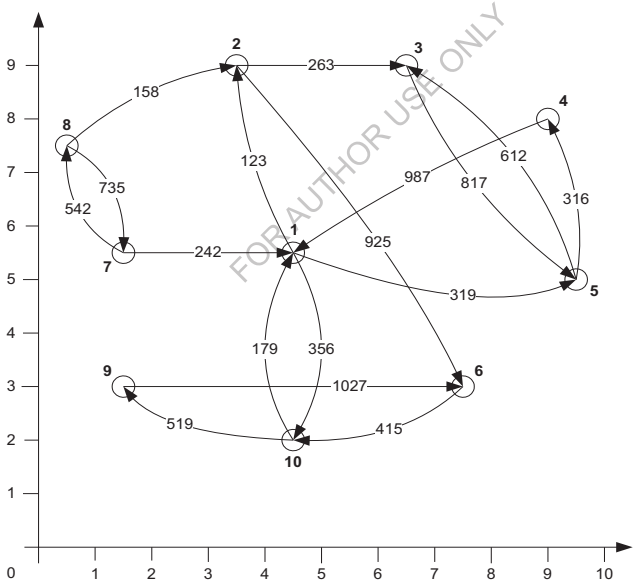


Рис. 3.9 Граф базовой модели

Скорость нахождения решения в зависимости от вероятности единичного бита выглядит, как показано на рисунках 3.10 и 3.11.

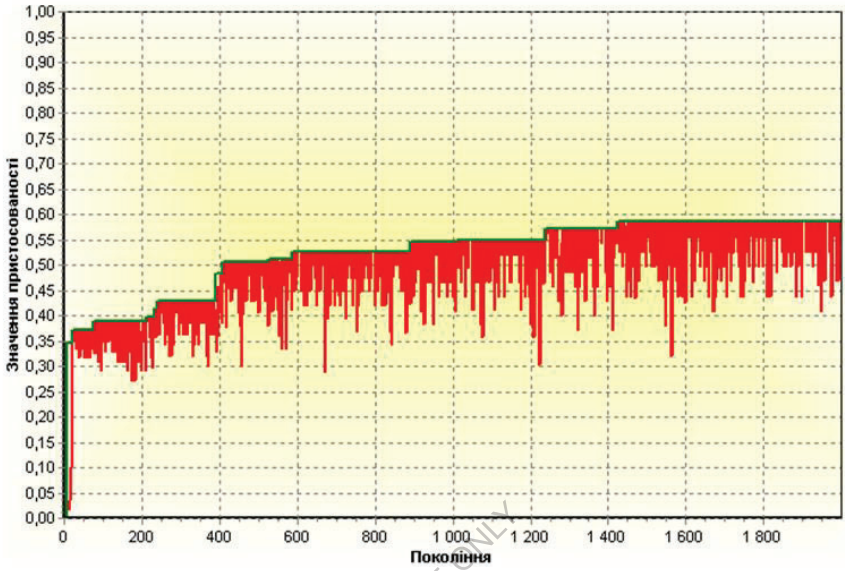


Рис. 3.10 Рост приспособленности при оптимальном расчёте вероятности единичного бита

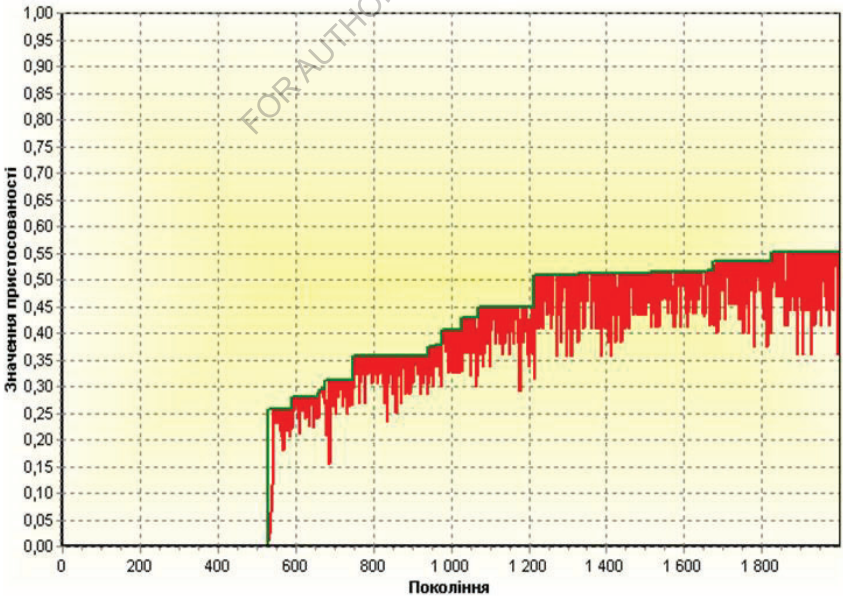


Рис. 3.11 Рост приспособленности при вероятности единичного бита равной 0,5

Для оценки скорости генетического алгоритма в зависимости от размерности задачи проведено сравнение времени поиска решения при разной размерности задачи. Особенностью опыта является то, что для каждой размерности поиск длился определенное количество поколений (10000 поколений). Такой подход позволяет оценить время исчисления фитнес-функции и выполнения генетических операторов в зависимости от размерности задачи, чем, в основном, и определяется скорость работы генетического алгоритма. Причем значение вероятности единичного бита при генерации начального поколения подбиралось оптимальным образом по формуле 3.23.

Для решения каждой задачи были приняты следующие параметры:

- размер популяции - 20;
- вероятность кроссинговера - 0,9;
- вероятность мутации - 0,1.

Всего было рассмотрено 5 различных задач:

- 6 узлов и 11 роутов (6N11F);
- 10 узлов и 17 роутов (10N17F);
- 15 узлов и 23 роута (15N23F);
- 20 узлов и 27 роутов (20N27F);
- 25 узлов и 41 роут (25N41F).

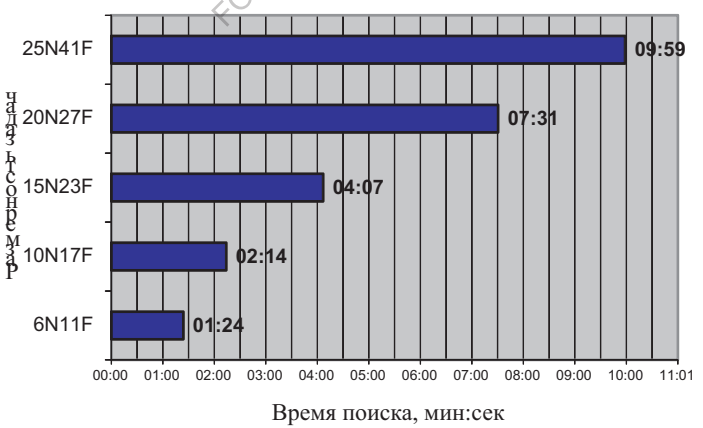


Рис. 3.12 Зависимость времени поиска от размерности задачи

Как и следовало ожидать, с ростом размерности задачи возрастает и время поиска. Это вполне закономерно, ведь с ростом размерности осложняется расчет функции приспособленности, а также растет размер хромосомы, замедляет работу генетических операторов. Во всех запусках без исключения было найдено хотя бы какое-то решение. Это свидетельствует о том, что даже при больших размерах задачи алгоритм дает приемлемое решение за достаточно короткое время по сравнению с алгоритмами полного перебора.

Большое влияние на скорость получения решения и его качество имеет также и вероятность мутации. Как известно, вероятность мутации определяет разнообразие популяции. Чем выше вероятность, тем разнообразнее популяция. Развитие не сосредоточивается вокруг какого-то решения, а продолжает поиск новых решений в других точках ландшафта функции.

С целью изучения влияния вероятности мутации на скорость эволюции решения проведен следующий опыт. Оценивалось значение функции приспособленности, достигнутое в течение 3000 поколений при различных вероятностях мутации. Так как генетические алгоритмы несут некоторый элемент случайности, то принималось во внимание среднеарифметическое значение достигнутой приспособленности на основе 5 запусков. Следует также заметить, что в исследуемой программе использовалось явление пятикратного увеличения вероятности мутации потомков при родства особей родительской пары.

Влияние вероятности мутации на скорость роста приспособленности проиллюстрировано на рис. 3.13.

Почти одинаковую приспособленность для вероятности мутации от 0,01 до 0,06 можно объяснить тем, что максимальная приспособленность, которая была получена при генерации начального поколения (использовался оптимальный показатель вероятности единичного бита согласно формуле 3.23) или в первые поколения развития, сохранила свое значение до конца процесса поиска решения.

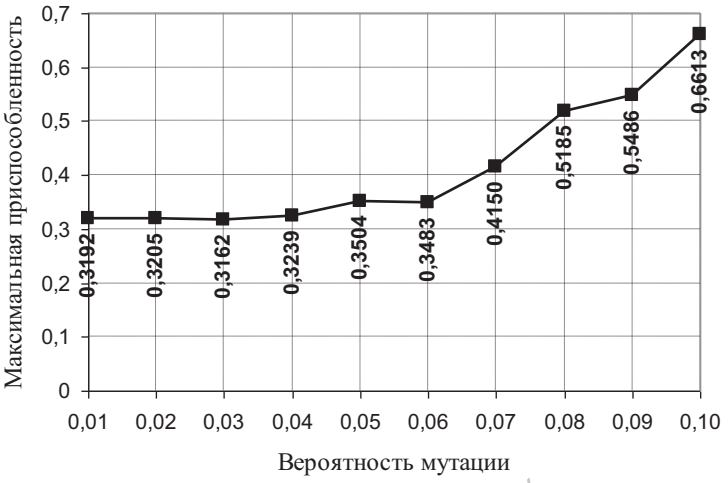


Рис. 3.13 Зависимость времени поиска от размерности задачи

3.3.2 Метод поиска рациональных "звездообразных" структур информационных систем с использованием кодирования Прюфера

В предыдущем п. 3.3.1 предложено расширение постановки задачи, описанной в п. 3.1 и модифицированный классический генетический алгоритм ее решения. Структуры, которые формируются при этом, имеют "звездообразный" вид, но система кодирования, используемая в алгоритме, охватывает все виды структур. Ограничение типов структур, а также введение в генетическом алгоритме цифр Прюфера для их кодирования, позволяет уменьшить время поиска рациональных решений. Кроме того, интересным для исследования является использование возможностей пакета MATLAB, его библиотеки GATool, для реализации предложенных алгоритмов. Это позволит использовать полученные результаты не только в процессе проектирования АСУ, но и в учебном процессе при изучении методов решения инженерных задач в MATLAB.

Рассмотрим алгоритмы реализации решения задачи.

Кодирования и декодирования решений задачи оптимизации. Для работы генетического алгоритма необходимо определиться с типом особей популяции, то есть с представлением решений задачи оптимизации.

При выборе способа кодирования скелетных деревьев необходимо учитывать, какой вид предположительно может иметь оптимальная структура ИУВМ. Если не учитывать ограничения на объем трафика в канале, то оптимальной структурой информационной системы будет структура «звезда» (рис. 3.14), при которой одна подсистема находится в центре «звезды», другие - «лучами» соединяются с центром.

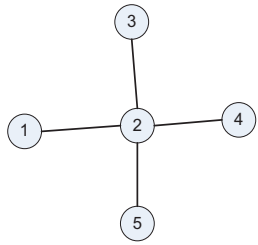


Рис. 3.14 Структура «звезда»

Оптимальность такой структуры в рамках данного метода обеспечивается минимальным приростом суммарного информационного потока из-за того, что дублирование информационных потоков в системе - минимальное. Если учитывать ограничения на объемы трафика, то, вероятнее всего, оптимальной структурой будет некоторая звездообразная структура, так называемый развитие «звезды» (рис.3.15).

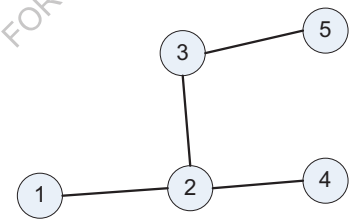


Рис. 3.15 " Звездообразная "структура

Для кодирования скелетных деревьев избран код Прюфера, который представляет собой способ однозначного кодирования дерева с помощью последовательности чисел. Согласно теоремы Келли [25], на N вершинах, что пронумерованы натуральными числами от 1 до N, существует ровно N^{N-2} различных скелетных деревьев. Прюфер доказал, что существует однозначное преобразование между скелетными деревьями с N вершинами и векторами длиной, в которых каждый элемент является натуральное число от 1 до N [200].

Алгоритм кодирования структуры остовного дерева приведен на рис. 3.16. В массиве code в конце работы алгоритма находится код Прюфера соответствующего дерева.

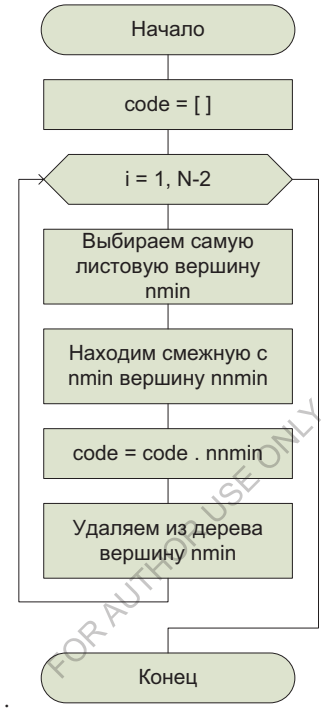


Рис. 3.16 Алгоритм кодирования деревьев кодом Прюфера

Рассмотрим пример кодирования (рис. 3.17).

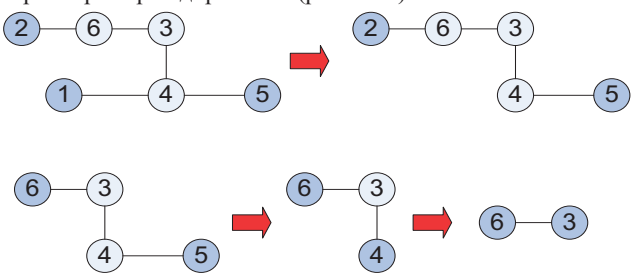


Рис. 3.17 Процесс кодирования дерева

Вершина 1 является самой листовой вершиной графа. Вершина 4 является смежной с вершиной 1. То есть, цифра 4 будет первой в коде Прюфера. Удаляем вершину 1 и ребро (1,4). Повторяем процесс, пока не останется ребро (3,6). Полученный код Прюфера например из рис. 3.17 - [4 6 4 3].

Алгоритм декодирования дерева из кода Прюфера приведена на рис. 3.18.

Рассмотрим пример декодирования кода Прюфера [3 2 2 1] (см. рис. 3.19).

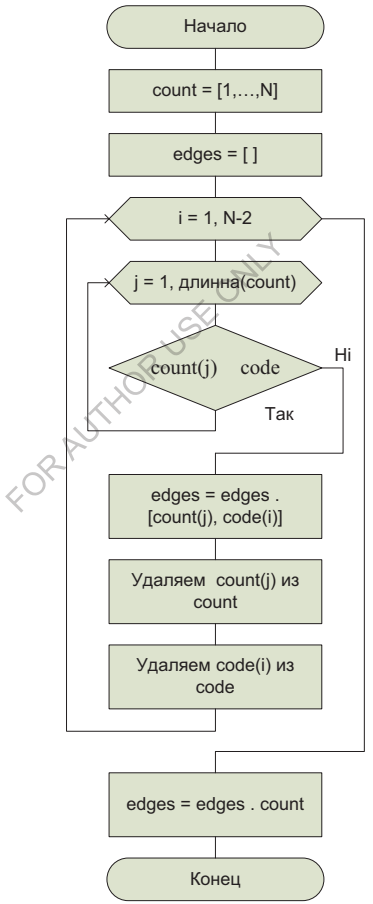


Рис. 3.18 Алгоритм декодирования деревьев числами Прюфера

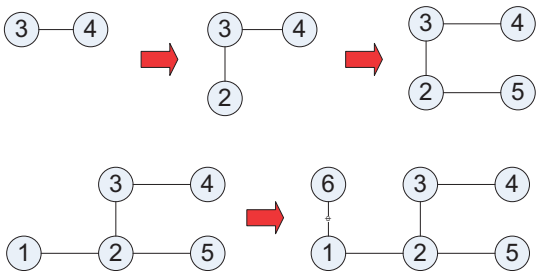


Рис. 3.19 Процесс декодирования дерева

Для кода Прюфера P = [3 2 2 1] набор вершин, что отсутствуют в коде – $\overline{P}$ = [4 5 6]. Вершина 4 является самой маленькой из $\overline{P}$, а вершина 3 является первой в P. Добавляем ребро (3,4) к дереву, удаляем вершину 3 из P и вершину 4 с $\overline{P}$. Поскольку вершина 3 теперь отсутствует в P, добавляем ее в $\overline{P}$. На следующем шаге имеем P = [2 2 1], $\overline{P}$ = [3 5 6]. Повторяем указанные выше действия, пока в P не останется элементов, а $\overline{P}$ = [1 6]. Добавляем к дереву последнее ребро (1,6). На этом процесс декодирования завершается.

Код Прюфера широко используется исследователями при решении MST-задач [185]. Данный код демонстрирует свою эффективность при поиске именно звездообразных скелетных деревьев. «Звезда» из N вершин в коде Прюфера представляет собой последовательность длиной, в которой все элементы равны. Например, последовательность [3 3 3] описывает «звезду» из 5 вершин с центральной вершиной 3. Такое свойство кода можно использовать для ускорения поиска оптимальной структуры при формировании первого поколения алгоритма.

Так как тип особей генетического алгоритма при использовании кода Прюфера - целочисленные строки, в MATLAB необходимо самостоятельно реализовать процедуры инициализации популяции, оператор кроссинговера и мутации.

Инициализация популяции. Процесс инициализации популяции проходит в начале работы генетического алгоритма. Реализация данной процедуры может значительно ускорить поиск оптимальной структуры, чем повышается эффективность всего алгоритма.

Несмотря на то, что оптимальным решением задачи оптимизации ИС зачастую является «звезда» или ее развитие, смысл в инициализации части начального поколения всех возможных «звезд» для заданного количества узлов. Другая часть популяции полностью случайной для внесения большей генетического разнообразия. Таким образом, если по условиям задачи, на узел не накладывают ограничения на трафик, а также размер популяции является достаточным для инициализации всех возможных «звезд», оптимальное решение будет найден уже на первой итерации генетического алгоритма.

Алгоритм инициализации популяции приведен на рис. 3.20.

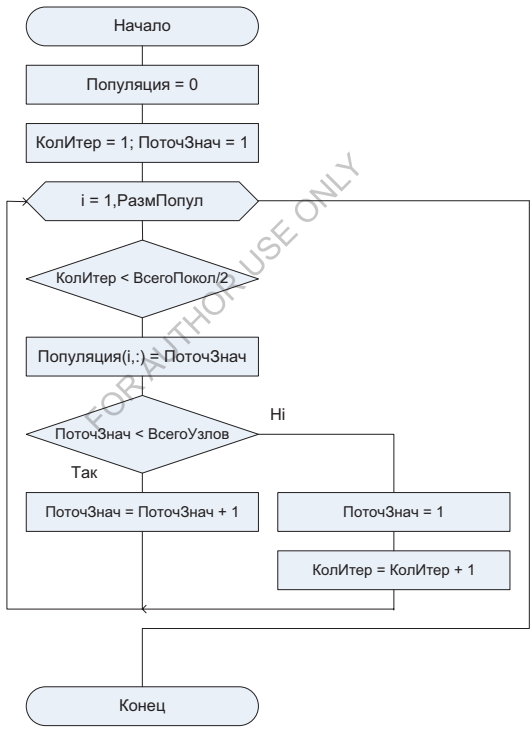


Рис. 3.20 Алгоритм инициализации популяции

Оператор кроссинговера. Через специфический тип особей генетического алгоритма был разработан специальный оператор кроссинговера. В [200] для решения задач MST (Minimum Spanning Tree) рекомендуется использовать

двухточечный кроссинговер, который является эффективнее одноточечного. Особенностью предлагаемой реализации кроссинговера является образование только одного потомка, причем порядок обмена родителей своими фрагментами является случайным.

Алгоритм реализации кроссинговера приведен на рис. 3.21.

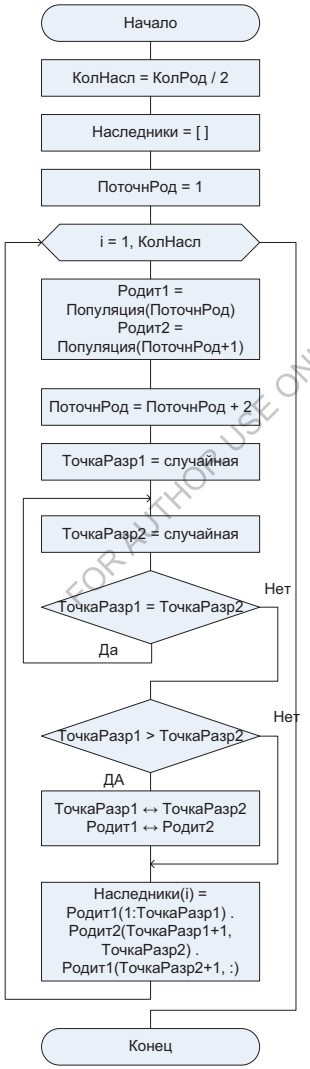


Рис. 3.21 Алгоритм оператора кроссинговера

Оператор мутации. Для более эффективной борьбы с преждевременной сходимостью, а также для внесения большего разнообразия в популяцию используем двухточечную мутацию. За основу взята стандартная двухточечная мутация для битовых особей, что предлагается утилитой GATool в пакете MATLAB, модифицированная для целочисленных строк. К тому же, если в течение пяти поколений алгоритм не улучшает решение, случайная особь в популяции мутирует полностью.

Алгоритм оператора двухточечной мутации приведен на рис. 3.22.

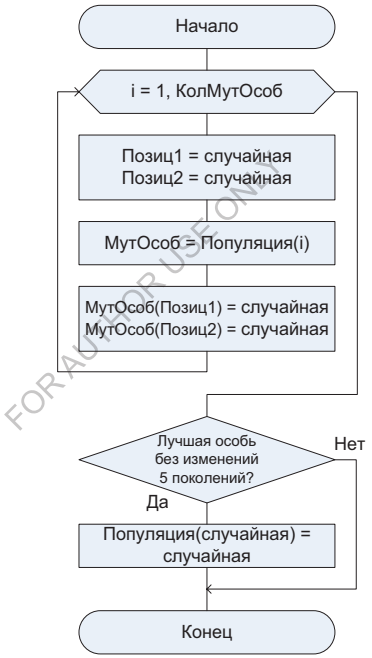


Рис. 3.22 Алгоритм оператора мутации

Функция приспособленности. Разработка функции приспособленности является неотъемлемой частью проектирования генетического алгоритма. Функция приспособленности должна давать объективную оценку эффективности каждой особи, учитывая все ограничения задачи.

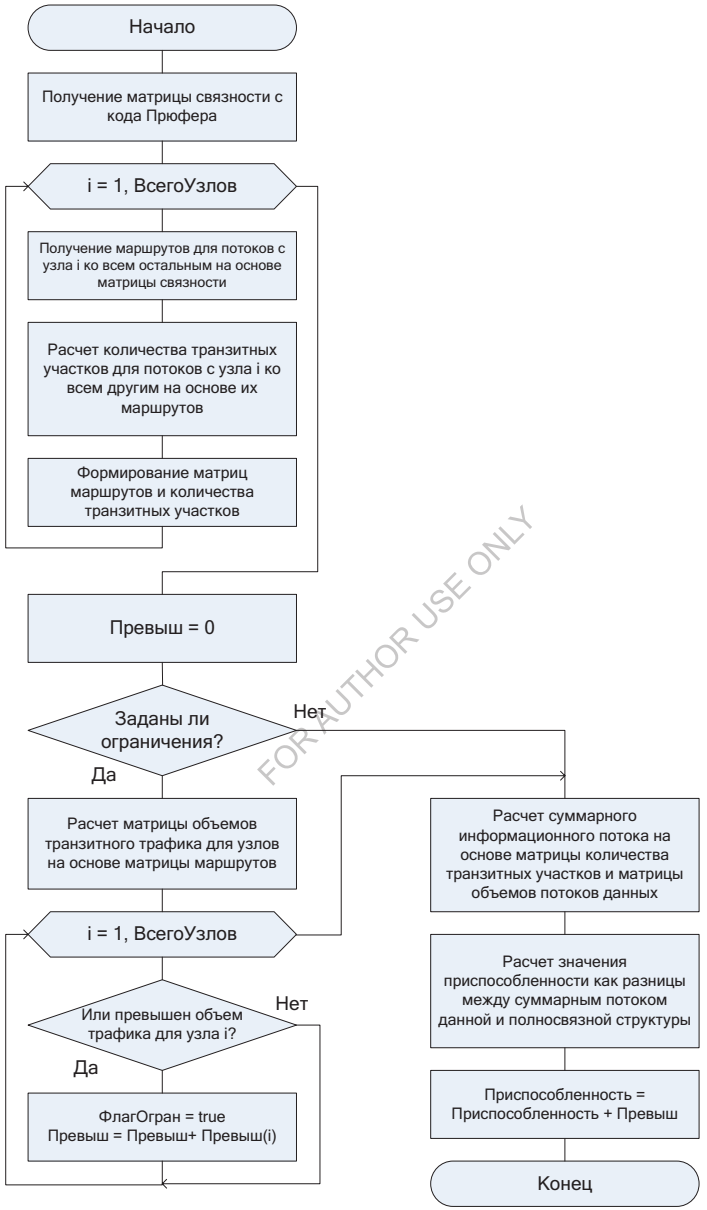


Рис. 3.23 Алгоритм получения функции приспособленности

В случае оптимизации структуры ИУВМ функция приспособленности должна вычислять прирост суммарного информационного потока в структуре, закодированная в особи популяции, в сравнении с полносвязной структурой системы. Функция должна расшифровать код Прюфера (т.е. особь популяции), рассчитывать маршруты для всех информационных потоков в системе, на основе которых рассчитывается прирост суммарного информационного потока, а также учитывать, выполняются ли ограничения на трафик в подсистемах.

Значение функции состоит из значения прироста суммарного информационного потока, а также из суммарного превышения установленных ограничений на трафик всех узлов системы. Таким образом, чем меньше значение функции приспособленности, тем лучшее решение предлагает особь популяции.

Алгоритм расчета функции приспособленности приведен на рис. 3.23.

Замечания по программной реализации решения задачи оптимизации. На основе алгоритмов, описанных выше, на языке программирования MATLAB была реализована программа GAOSIS, выполняет оптимизацию структуры информационной системы. Входными данным для нее матрица потоков данных и матрица ограничений трафика. Результатом работы программы является набор ребер графа. На рисунке 3.24 представлена визуализация итоговой структуры системы.

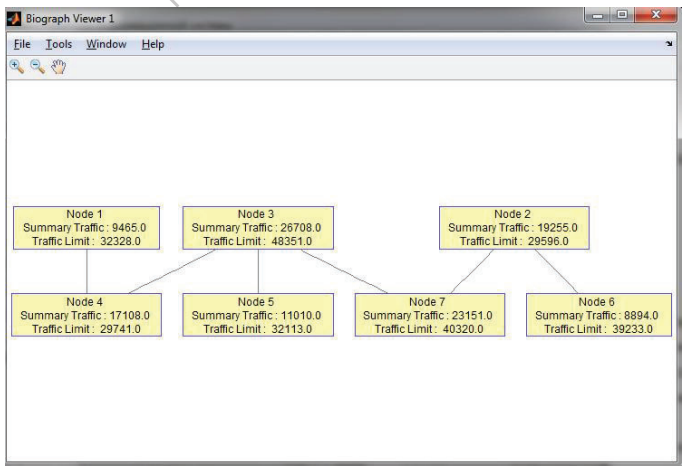


Рис 3.24 – Визуализация решения задачи оптимизации структуры

3.4 Ресурсосберегающие методы выбора технических структур децентрализованных информационно-управляющих систем

3.4.1 Метод исследования условий целесообразности децентрализации функций управления в системах горочной автоматики

Развитие систем автоматизации сортировочных горок в Украине и за рубежом происходило от создания первых централизованных компьютерных систем на базе мини-ЭВМ (СМ-2) к построению функционально распределенных систем на основе промышленных микро-ЭВМ и микроконтроллеров (микро-ДАТ, СМ-1800/1810, микроконтроллеры Advantech) [69, 158]. С переходом на децентрализованные (иерархические) системы при разнообразии вариантов распределенных структур нерешенным остался вопрос обоснованности перехода к децентрализованному управлению. Несмотря на многочисленные попытки найти ответ на данный вопрос [24, 47, 108, 150] он остается актуальным в условиях развития технических средств автоматизации сортировочных горок.

Целью данной работы является исследование условий целесообразности создания децентрализованных систем. Эта задача является актуальной и для крупной интегрированной автоматизированной системы управления грузовыми перевозками УКРЗАЛІЗНИЦІ - АСУ ГП УЗ-Е [69].

При рассмотрении вопросов оптимизации структуры иерархической системы мы будем считать, что сортировочная горка, как технологический объект управления сортировочной станцией, имеет локальные устройства автоматики.

Централизованная система управления, оптимизируя в целом работу сортировочной горки, корректирует работу местных устройств автоматики, формируя уставки для регуляторов (например, тормозных позиций, стрелочных переводов). По такому принципу была построена система АСУ РСГ для станции Ясиноватая Донецкой железной дороги [69].

С ростом мощности управляющих вычислительных машин (УВМ) возникает интерес рассмотреть целесообразность передачи функций местных устройств автоматики управляющим ЭВМ более высокого ранга, которые реализуют алгоритм оптимизации режима работы системы [150].

Если ранее возможности систем горочнуй автоматики были ограниченными с точки зрения реализации сложных алгоритмов управления, а

УВМ не отличались большим быстродействием, то сейчас есть технические возможности наращивать мощности как микроконтроллеров, так и управляющих ЭВМ. Поэтому передача функций местных регуляторов на УВМ существенно не повлияет на ее мощность (объем памяти, быстродействие и т.д.).

Однако такая централизация управления существенно повышает ответственность УВМ. Очевидно, что выход из строя УВМ нарушает управления целой сортировочной горкой, что приводит почти всегда к большим экономическим потерям. Поэтому многие авторы считают [126, 128, 144], что определяющим в решении этого вопроса является надежность УВМ.

Сортировочная горка О состоит из n управляемых объектов, состояние каждого из которых O_i можно охарактеризовать двумя параметрами: y_i - регулируемый параметр; x_i – управляющее действие (z_j - контролируемый, f_g - неконтролируемый параметры, см. рис. 3.25).

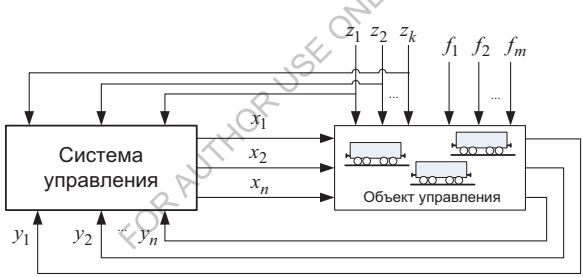


Рис. 3.25 Сортировочная горка как объект управления

Таким образом, состояние системы в каждый момент времени, может быть охарактеризовано векторами: состояния регулируемого параметра $Y=\{y_1,...,y_n\}$; положения регулирующего органа $X=\{x_1,...,x_n\}$ и вектор-уставкой $Y'=\{y'_1,...,y'_n\}$. Вектор-уставка Y' представляет собой задачу на поддержку регулируемого параметра и в общем случае может отличаться от его истинного значения - вектора *Y*.

Если система не имеет локальных регуляторов, то УВМ непосредственно формирует вектор X и выдает его на исполнительные органы. В данном случае мы имеем централизованную систему (см. рис. 3.30). При наличии местных регуляторов (МГ), реализующие функции локального управления, функции

УВМ упрощаются к формированию только уставок Y' для МР. В этом случае говорят об иерархической структуре системы управления (см. рис. 3.26).

Пусть система состоит из УВМ и *n* локальных подсистем.

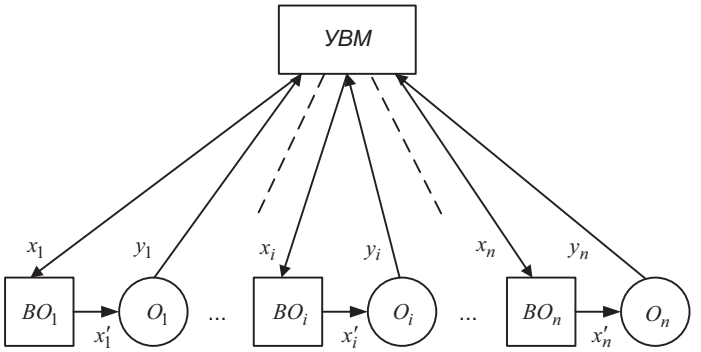


Рис. 3.26 Структура централизованной системы управления горкой с применением УВМ без автономных подсистем управления.

На рис. 3.26 и 3.27 приемлемые следующие обозначения: УВМ - управляющая вычислительная машина; МР - местный регулятор; O_i – *i*-а подсистема; ИО – исполнительный орган.

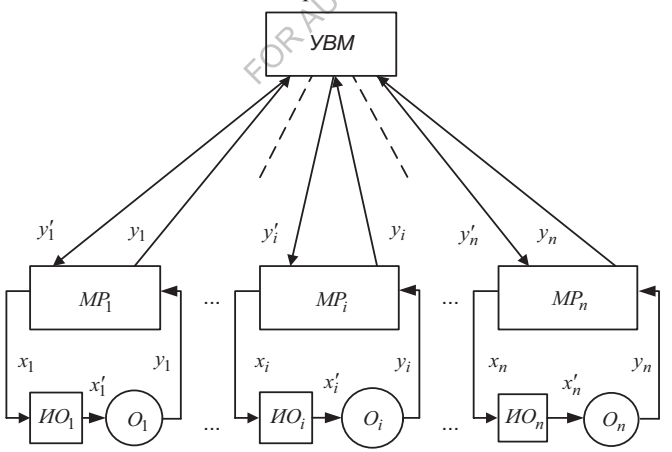


Рис. 3.27 Структура иерархической системы с применением УВМ и местных подсистем управления

Реализация алгоритма управления сводится к минимизации (максимизации) функции вида

$$B = \varphi(Y, X) \tag{3.24}$$

(например, минимизации отклонения скорости выхода из тормозной позиции от заданной по алгоритму управления).

Оптимизация предполагает поиск такого оптимального значения вектора управления *U**, чтобы функционал управления (3.24) при данном X принимал оптимальное значение

$$B^* = \min(\max)\varphi(Y, X) \tag{3.25}$$

или

$$B^* = \varphi\left(Y, X^*\right). \tag{3.26}$$

Оценку вариантов структур будем проводить по критерию полных затрат ресурсов. Для упрощения расчетов принимаем, что все *n* контуры управления одинаковы, расходы на эксплуатацию систем также одинаковы и ненадежностью исполнительных органов можно пренебречь.

Управление системой будем представлять в режиме разделения времени следующим образом. По схеме на рис. 3.30 (централизованная система управления) УВМ через интервалы времени $T_{\text{ц}}(T_{\text{ц}}, 2T_{\text{ц}}, ..., kT_{\text{ц}})$ циклически получает информацию о состоянии управляемого объекта в виде векторов *Y*. В результате переработки информации по алгоритму (3.25) УВМ осуществляет расчет новой уставки Y' и выдачу команды управления на исполнительные органы - вектор положения *X*.

По схеме на рис. 3.27 (иерархическая система управления) УВМ также получает информацию в виде векторов *Y*. Однако в этом случае УВМ выдает только вектор-уставку Y'. Расчет вектора *X* осуществляют местные регуляторы МР; УВМ только координирует их работу, корректируя уставку-вектор Y'.

Посмотрим, что произойдет в случае отказа аппаратуры в k-й интервал времени.

При выходе из строя УВМ в централизованной структуре возможны следующие ситуации:

1) немедленно прекращается работа сортировочной горки. Потери определяются простоями объекта и могут быть оценены в полной потерей

эффективности за время устранения неисправности (восстановление системы) $T_{\text{вост}}$:

$$p_0 = B^* T_{\text{вост}}.$$

2) Некоторое время после отказа УВМ система продолжает функционировать нормально и затем происходит нарушение ее работы. В этом случае поведение объекта может изменяться в зависимости от того, в каком положении окажутся исполнительные органы. Они могут: а) оставаться в том положении, в котором они были в момент аварии t_0, то есть $X = X_0$;

б) переводится в аварийное состояние, характеризующееся значениями $X = X_{\text{ав}}$.

Во второй ситуации ущерб p_1 будет состоять из двух частей (см. рис.3.28), $y_i = v_i$- скорость выхода из *i*-й тормозной позиции, $v_{i\min} < v_i(t) < v_{i\max}$). На участке t_1 он пропорционален разности между оптимальным и реальным значениями эффективности:

$$p_{11} = (B^* - B_1)t_1. \qquad (3.27)$$

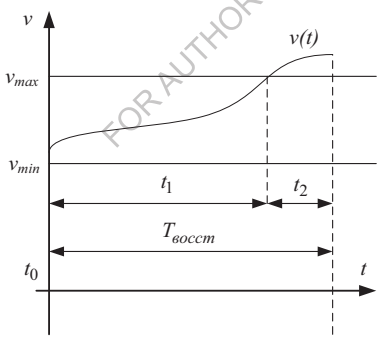


Рис. 3.28 Пример поведения регулируемого параметра *v(t)* в пределах *($v_{мин}$, $v_{макс}$)*.

На участке t_2 происходит выход параметра $v_i(t)$ за допустимые пределы $v_i(t) > v_{i\max}$ и местную защиту отключает объект, то есть потери на этом участке p_{12} определяются полной потерей эффективности:

$$p_{12} = B^* t_2.$$

Полные потери для второй ситуации

$$p_1 = p_{11} + p_{12} = (B^* - B_1)t_1 + B^* t_2 \ ,$$

где

$$T_{восст} = t_1 + t_2; \lim_{t_1 \to \infty} p_1 = B^* T_{восст} \ .$$

B_1 - эффективность при не оптимальном управлении в централизованной системе (ручное управление объектами сортировочной горки).

В иерархической системе при отказе УВМ управления теряется не полностью, снижается только его эффективность. Пусть в результате отказа она принимает значение B_2. В этом случае ущерб от потери оптимального управления

$$p_2 = (B_2^* - B_2)T_{\text{вост}} . \qquad (3.28)$$

Этот ущерб, как правило, значительно меньше ущерба, получаемого при выходе из строя УВМ в централизованной системе. Уменьшение ущерба в системе с иерархической структурой достигается за счет увеличения ее стоимости из-за наличия локальных регуляторов. Кроме того, эффект уменьшения потерь снижается за счет дополнительных потерь, вызываемых ненадежностью локальных регуляторов. Примем для дальнейшего анализа следующие

Предположение:

B^* - потери в единицу времени, когда горка не работает;

B_1 - потери в единицу времени, когда в централизованной системе не работает УВМ (неоптимальное управления);

B_2 - потери в единицу времени, когда в иерархической системе не работает УВМ (неоптимальное управления);

b_i^* - потери в единицу времени, когда не работает i-ый контур управления,

b_i - потери в единицу времени при неоптимальной значении эффективности i-го контура.

$$B^* > B_1 > B_2 \gg b_i^* > b_i .$$

Тогда полные затраты для централизованной P_1 и иерархической P_2 систем можно оценить следующим образом:

$$\mathrm{P}_1 = \left(B^* - B_1\right)\lambda_0 t_1 + B^* \lambda_0 t_2 + C_0^{\text{ц}}, \qquad (3.29)$$

$$\mathrm{P}_2 = (B^* - B_2)\lambda_0 T_{\text{вост}} + \sum_{i=1}^{n}\left[(b_i^* - b_i)\lambda_i t_{i1} + b_i^*\lambda_i t_{i2}\right] + C_0^{\text{дц}} + \sum_{i=1}^{n} c_i\,, \quad (3.30)$$

где λ_0 - интенсивность отказов УВМ в иерархической и централизованной системах; λ_i - интенсивность отказов i-го локального регулятора; $C_0^{\text{ц}}, C_0^{\text{дц}}, c_i$ - стоимость УВМ соответственно в централизованной системе, в децентрализованной (иерархической) системе и стоимость микроконтроллерной системы i-го локального регулятора.

Будем считать, что

$$C_0^{\text{ц}} > C_0^{\text{дц}} \gg c_i\,.$$

Для упрощения дальнейшего анализа принимаем, что система однородна по своему составу, то есть

$$b_i = b_n;\ \lambda_i = \lambda_n;\ c_i = c_n;\ i = 1,2,...,n\,. \quad (3.31)$$

Для *i*-го контура управления с учетом возможного выхода из строя регулятора по аналогии с централизованной системой управления (см. рис. 3.28) можно написать

$$t_{i2} = T_{\text{восст } i} - t_{i1}\,,$$

где t_{i2} - время, в течение которого *i*-й параметр находится в допустимыми значениями; $T_{\text{восст } i}$ - время ремонта *i*-го регулятора; t_{i1} - время, в течение которого *i*-й параметр не выходит за допустимые пределы.

В первом приближении примем, что

$$\begin{aligned} &t_{i1} = t_{n1} = t_1,\ i = 1,2,...,n; \\ &T_{\text{восст}}^{\text{ц}}(\text{КУВМ}) = T_{\text{восст}}^{\text{дц}}(\text{КеОМ}) = T_{\text{восст}}(c_n). \end{aligned} \quad (3.32)$$

Для дальнейшего исследования введем коэффициент

$$t_1 = rT_{\text{восст}},\ t_2 = (1-r)T_{\text{восст}}\,.$$

Тогда из (3.29) и (3.30) имеем

$$\mathrm{P}_1 = \left(B^* - B_1\right)\lambda_0 r T_{\text{восст}} + B^*\lambda_0(1-r)T_{\text{восст}} + C_0^{\text{ц}}\,, \quad (3.33)$$

$$\mathrm{P}_2 = (B^* - B_2)\lambda_0 T_{восст} + (B^* - B_1)\lambda_n r T_{\text{восст}} + B^*\lambda_n(1-r)T_{\text{восст}} + C_0^{\text{дц}} + nc_n. \quad (3.34)$$

Перейдем к интегральному показателю надежности - коэффициента готовности

$$K_{\text{г}}^{\text{ц}}(\text{КУВМ}) = K_{\text{г}}^{\text{дц}}(\text{КУВМ}) = K_0 = \frac{T_{\text{но}}}{T_{\text{но}} + T_{\text{восст}}} = \frac{1/\lambda_0}{1/\lambda_0 + T_{\text{восст}}}, \quad (3.35)$$

$$K_{\text{г}}(c_n) = K_n = \frac{T_{\text{но}(n)}}{T_{\text{но}(n)} + T_{\text{восст}}} = \frac{1/\lambda_n}{1/\lambda_n + T_{\text{восст}}}, \qquad (3.36)$$

где $T_{\text{но}}$ - среднее время наработки на отказ.

Откуда

$$\lambda_0 = \frac{1 - K_0}{K_0 T_{\text{восст}}}; \;\; \lambda_n = \frac{1 - K_n}{K_n T_{\text{восст}}}. \qquad (3.37)$$

В дальнейшем введем коэффициент α:

$$K_n = \alpha K_0. \qquad (3.38)$$

Подставим (3.37) и (3.38) в (3.33), (3.34), нормализуем полные затраты и введем обозначения

$$\delta_1 = \frac{B^* - B_1}{B^*}; \;\; \delta_2 = \frac{B^* - B_2}{B^*}.$$

В результате получим

$$\tilde{\text{P}}_1 = \frac{\text{P}_1}{B^*} = \frac{(1 - K_0)(1 + r(\delta_1 - 1))}{K_0} + \frac{C_0^{\text{ц}}}{B^*}, \qquad (3.39)$$

$$\tilde{\text{P}}_2 = \frac{\text{P}_2}{B^*} = \frac{\delta_2(1 - K_0)}{K_0} + \frac{(1 - \alpha K_0)(1 + r(\delta_1 - 1))}{\alpha K_0} + \frac{C_0^{\text{дц}} + nc_n}{B^*}, \qquad (3.40)$$

На рисунке 3.29 построены графики $\tilde{\text{P}}_1 = f_1(K_0)$ и $\tilde{\text{P}}_2 = f_2(K_0)$ при следующих предположениях $r = 0,9$ и $\alpha = 1,1$.

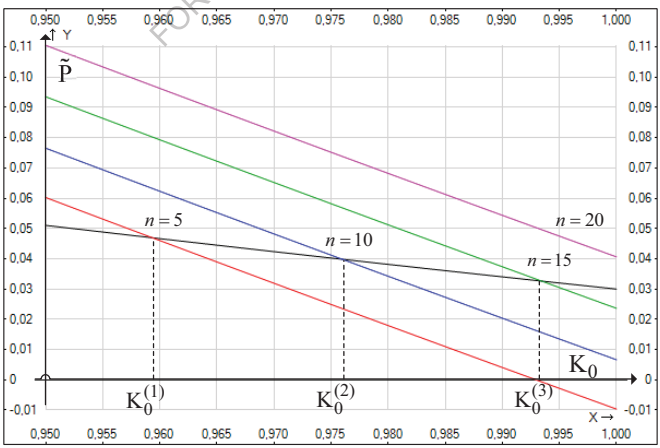


Рис. 3.29 Зависимость изменения эффективности систем от надежности УВМ и локальных регуляторов на микроконтроллерах

Из графиков следует, что с ростом коэффициента готовности потери в иерархической децентрализованной системе по сравнению с централизованной уменьшаются ($K_0^{(1)}, K_0^{(2)}, K_0^{(3)}$-точки на графике, где потери в обеих системах одинаковы). Из (3.40) следует, что потери в иерархической системе линейно зависят от числа и стоимости локальных микроконтроллерных регуляторов. При заданных показателях надежности (K_0, α)и стоимости подсистем в технической структуре системы управления ($C_0^{\text{ц}}, C_0^{\text{дц}}$ и c_n) число контуров локального управления n может быть выбрано на основании решения уравнений (3.39) и (3.40).

Если сравнить (3.39) и (3.40), и ввести нормализованые коэффициенты

$$a_1 = 10^{-3}\delta_2 B^*;\ a_2 = 10^{-3} c_n;\ a_3 = 10^{-3}(C_0^{\text{дц}} - C_0^{\text{ц}}), \qquad (3.41)$$

можем получить необходимые уравнения двух переменных K_0 и n:

$$a_1(1 - K_0)/K_0 + a_2 n + a_3 = 0.$$

На рисунке 3.30 представлены плоскости, отражающие геометрическое место точек $K_0^{(i)}$, в которых потери в централизованной и иерархической (децентрализованной) системах одинаковы и соответствует изменению приоритетов в построении систем в зависимости от коэффициентов готовности и количества локальных регуляторов. Анализ полученных результатов показывает, что рациональные решения расположены в зоне высоких коэффициентов готовности и небольшого количества локальных подсистем. Зона рациональных решений увеличивается, если уменьшается значение коэффициентау a_1. Количество локальных контуров не превышает 15 подсистем.

Следует учитывать, что полученные результаты справедливы только для случая конечных и аддитивных потерь. Для объектов, полный отказ устройств автоматики которых может привести к авариям типа катастроф или связана с возможностью человеческих жертв, выбор несомненно должен быть решен в пользу комбинированной системы.

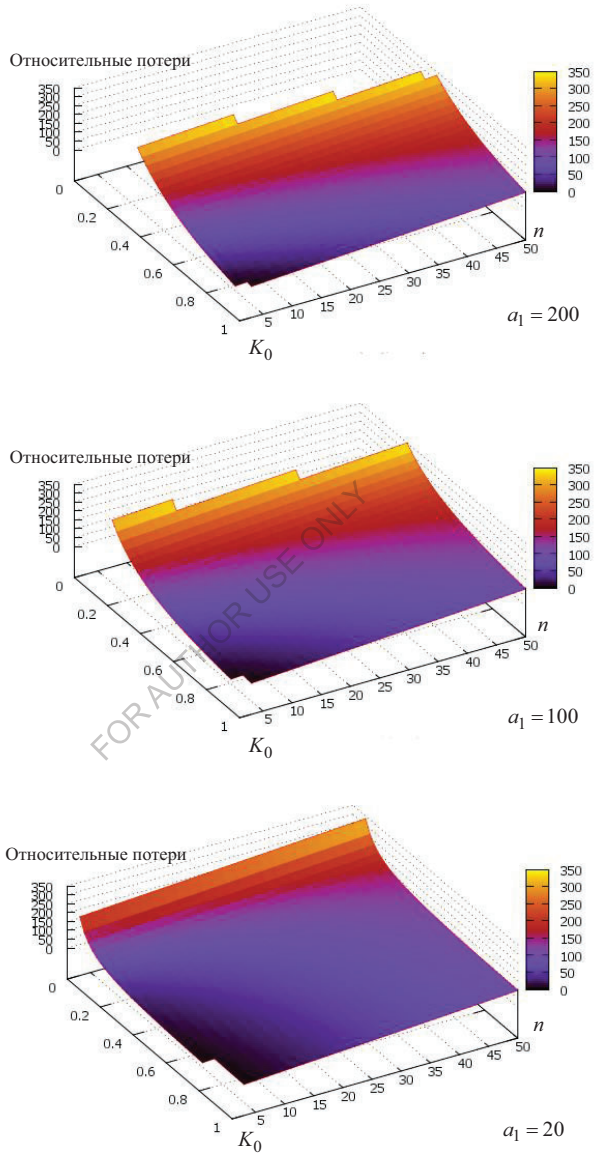


Рис. 3.30 - Плоскости равных потерь
централизованных и иерархических систем

3.4.2 Метод выбора технических структур цифровых управляющих систем (ЦУС) по критерию эффективного использования вычислительных ресурсов

В современных и перспективных системах автоматизации управления транспортными процессами на каждом этапе их развития решается вопрос декомпозиции системы и представление ее или в виде централизованной структуры, или иерархической системы, для которой необходимо выбрать рациональную степень декомпозиции [69]. Сложность этой задачи состоит в том, что на ранних стадиях проектирования отсутствуют многие параметры технических средств автоматизации, что требует введения ряда частных критериев и экспертных оценок. В работе [150] рассматриваются условия целесообразности передачи в иерархической системе функций местных устройств автоматики управляющим устройствам более высокого уровня, где используются многоуровневые оценки ущерба в зависимости от режимов функционирования технологической системы и соответствующих относительных потерь, оценки стоимости центральных управляющих ЭВМ и микроконтроллеров в локальных регуляторах, а также интенсивности отказов и интенсивности восстановления используемых устройств. Развитием описанного подхода являются модели, предложенные в [65], где одним из параметров выступает коэффициент готовности и приводятся экспертные оценки потерь и стоимости технических средств. В работе [189] предлагаются относительные оценки и сравнения надежности централизованных и децентрализованных АСУ ТП, где для каждого состояния вводится весовой коэффициент потерь и рассчитываются средние потери. Имитационно-аналитический подход предлагается в [213], а в [211] используются наборы частичных качественных критериев и ограничений для выбора централизованной или децентрализованной системы автоматизации. Для выбора варианта структуры децентрализованной микропроцессорной системы управления сортировочной горкой в [116] впервые предложен набор относительных коэффициентов эффективности увеличения стоимости АСУ по 5 частичным и одному общему показателям.

Описанные подходы не учитывают сложности автоматизированных процессов управления, которые определяют необходимые вычислительные ресурсы системы и, в конечном итоге, ее стоимость (см. рис. 3.31).

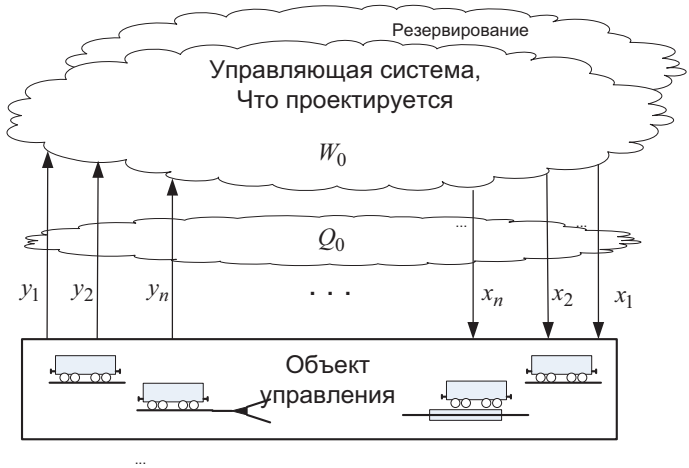


Рис. 3.31 Система, что проектируется с нужной вычислительной мощностью W_0 и информационными потоками Q_0 (централизованная структура)

Необходимые вычислительные ресурсы обычно выражаются в виде суммарной вычислительной мощности W (производительности, быстродействия в MIPS - в миллионах команд в секунду) всех компьютеров в системе. Для оценки стоимости вычислительных ресурсов используется известный закон Гроша

$$C_0 = k_0 W^{g_1}, \qquad (3.42)$$

где C_0 - стоимость ресурсов (дол.), k_0- коэффициент пропорциональности, $g_1 = 0{,}5$, W - производительность в MIPS.

Несмотря на солидный возраст закона [190], он продолжает действовать [192], [197]. По утверждению Эйн-Дора [181] он хорошо работает в семействе микроконтроллеров и управляющих ЭВМ (например, Advantech [158]). В работах [144, 182] предлагаются подходы к выбору концепции управления с помощью микроконтроллеров и структуры систем автоматизации на основе описанного закона.

В данной работе, в развитии [182], предлагается методика, аналитические и графические модели и методы для выбора технических структур автоматизации сортировочных горок по интегральному критерию оценки вариантов структур с учетом стоимости необходимых вычислительных и

коммуникационных ресурсов, потерь на диспетчеризацию и изменения потоков в каналах, а также организацию резервирование.

Рассмотрим основные предположения. В настоящее время доказана на практике преимущество использования в АСУТП децентрализованных управляющих систем иерархического типа по сравнению с централизованным управлением [211, 69, 144]. При этом считается следующее.

1) Цифровые управляющие системы (ЦУС) иерархического типа, которые содержат n специализированных локальных микроконтроллерных подсистем $F_i,\ i=\overline{1,n}$ и одну универсальную подсистему верхнего уровня F_{11}, являются более живучими по сравнению с централизованной системой (рис. 3.32).

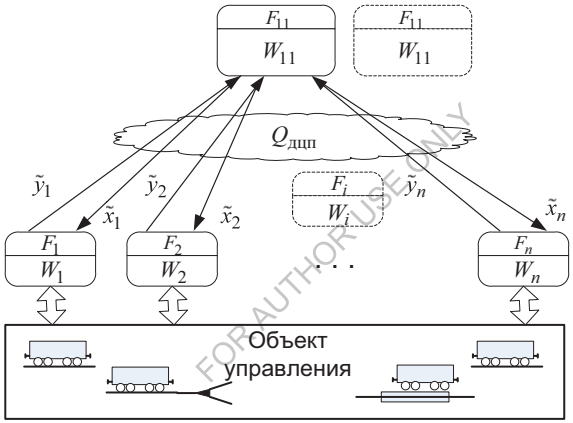


Рис. 3.32 Децентрализованная иерархическая ЦУС

2) Предварительная обработка данных в локальных подсистемах сокращает потоки информации в единицу времени между уровнями в управляющей системе до значения $Q_{дцп}$.

$$Q_0 = Y \cup X,\ Y = \{y_1, y_2, ..., y_n\},\ X = \{x_1, x_2, ..., x_n\}, \tag{3.43}$$

$$Q_{дцп} = \tilde{Y} \cup \tilde{X},\ \tilde{Y} = \{\tilde{y}_1, \tilde{y}_2, ..., \tilde{y}_n\},\ \tilde{X} = \{\tilde{x}_1, \tilde{x}_2, ..., \tilde{x}_n\}, \tag{3.44}$$

$$Q_{дцп} < Q_0, \tag{3.45}$$

$$W_0 = W_{11} + \sum_{i=1}^{n} W_i\,. \tag{3.46}$$

3) Наличие локальных подсистем снижает нагрузку на управляющую ЭВМ верхнего уровня не только за счет перераспределения функций, но и за счет уменьшения расходов на диспетчеризацию задач.

4) В децентрализованной системе должны использоваться унифицированные проектные решения, благодаря чему сокращаются затраты на проектирование и эксплуатацию таких систем.

Значения характеристик W_0 *i* Q_0 можно определить на основе анализа информационной модели функций, что автоматизируются, в соответствии с методикой [78].

Рассмотрим модели для сравнения структур ЦУС по критерию затрат на обработку данных. В качестве основной альтернативы централизованной системы управления (рис. 3.31) будем рассматривать структуру, представленную на рис. 3.32.

Нужная вычислительная мощность W_0 для управления объектом автоматизации распределяется между уровнями управления с помощью коэффициента децентрализации $k_д$. Для центральной подсистемы $W_{11} = k_д W_0$, для всех *n* подсистем нижнего уровня - $(1-k_д)W_0$.

Расходы на подсистему зависят от ее вычислительной мощности и вычисляются по степенному закону

$$C_0 = k_0 W_0^{g_1};\ \ g_1 = 0,5. \qquad (3.47)$$

Стоимостной коэффициент k_0 зависит от типа решаемых задач (научно-технические задачи или экономические задачи бизнес-систем) и находится в пределах $k_0 = 1 \pm 0,005$ [197]. Для технических систем в целом $0,5 \le g_1 \le 0,8$, а для вычислительных машин - $g_1 = 0,5$ [212].

Для сравнительной оценки структур будем использовать соотношение или показатель эффективности обработки данных

$$E_о = \frac{C^{дц}}{C^{ц}}. \qquad (3.48)$$

Здесь, для централизованной системы,

$$C^{ц} = k_1 W_0^{g_1}. \qquad (3.49)$$

Для децентрализованной системы мощность подсистемы верхнего уровня будет составлять $k_д W_0$, а другая мощность $(1-k_д)W_0$ будет равномерно

распределена между n подсистемами нижнего уровня как $(1-k_д)k_{дi}W_0$. Тогда получим

$$C^{дц} = k_1(k_д W_0)^{g_1} + \sum_{i=1}^{n} k_2[(1-k_д)k_{дi}W_0]^{g_1}. \qquad (3.50)$$

Предполагается, что k_1 для центральных подсистем больше, чем k_2 для децентрализованных подсистем. Поскольку мощность между n подсистемами нижнего уровня распределяется равномерно, то $k_{дi} = \frac{1}{n}$, $i = \overline{1,n}$.

Приняв, что $g_1 = 0,5$; $k_{21} = \frac{k_2}{k_1}$ и подставив (3.49) і (3.50) в (3.48) получим

$$E_o = \frac{C^{дц}}{C^{ц}} = k_{21}\sqrt{n}\sqrt{1-k_д} + \sqrt{k_д}. \qquad (3.51)$$

Обозначив $k_{21}\sqrt{n} = y$, получим график плоскости, разделяющей области преимущества децентрализованных-централизованных структур ЦУС (рис. 3.33). Считая, что $k_д$ - малое, то есть основная нагрузка переносится на нижний уровень ЦУС, и $k_{21}\sqrt{n}$ - небольшое, получаем $E_0 < 1$ - децентрализация лучше. При уменьшении предварительной обработки $(k_д \uparrow)$ $E_0 \approx 1$. При большом количестве подсистем в децентрализованной ЦУС $(k_{21}\sqrt{n} \uparrow)$ - $E_0 > 1$. В то же время перенос всей обработки на нижний уровень $(k_д \downarrow)$ приводит к росту затрат.

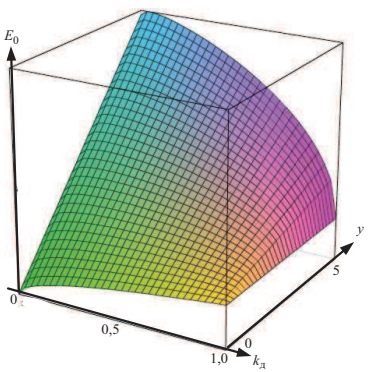


Рис. 3.33 Определение области преимущества децентрализованной обработки данных $(E_0 < 1)$

На рисунке 3.34 приведена предельная кривая равных затрат, при которой $E_0 = 1$. В этом случае (3.51) можно преобразовать в

$$y = \frac{1 - \sqrt{k_д}}{\sqrt{1 - k_д}}. \qquad (3.52)$$

Полученная кривая разделяет области преимущества структур ЦУС по затратам на обработку данных.

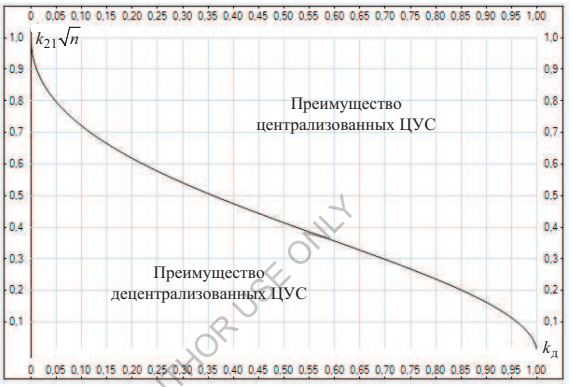


Рис. 3.34 Области преимущества структур обработки данных

Теперь рассмотрим модели сравнения структур ЦУС при условии суммарных затрат на обработку и передачу данных.

В предыдущей модели учитывались только расходы на обработку данных. Однако мы отмечали, что меняются и потоки данных между уровнями в ЦУС (см. рис. 3.32) и соответствующие расходы. Расходы на передачу данных в централизованных ЦУС можно оценить следующим образом [210]:

$$C_п^ц = \beta Q_0^{g_2}. \qquad (3.52)$$

Для децентрализованных систем

$$C_п^{дц} = \beta(f_{дцп} Q_0)^{g_2} = \beta(Q_{дцп})^{g_2}. \qquad (3.53)$$

В (3.52) і (3.53) - β, g_2 постоянные коэффициенты, $f_{дцп}$ - коэффициенты понижения информационных потоков при переходе на децентрализованную обработку. В дальнейшем будем считать, что $f_{дцп} = 0{,}1$, то есть объемы потоков информации уменьшаются в 10 раз. Кроме того, введем коэффициент

$$k_{\text{п/о}}^{\text{ц}} = \frac{\beta Q_0^{g_2}}{k_0 W_0^{g_1}}, \qquad (3.54)$$

который описывает отношение затрат на мощности передачи к расходам на мощности обработки в централизованной ЦУС. По данным [144], для централизованной системы $k_{\text{п/о}}^{\text{ц}} = 4$. В результате, предполагая, что $g_1 = g_2 = 0{,}5$, и основываясь на ранее принятых предположениях, можно записать

$$E_{\text{о/п}} = \frac{C^{\text{дц}} + C_{\text{п}}^{\text{дц}}}{C^{\text{ц}} + C_{\text{п}}^{\text{ц}}} = \frac{k_{21}\sqrt{n}\sqrt{1-k_{\text{д}}} + \sqrt{k_{\text{д}}} + k_{\text{п/о}}^{\text{ц}}\sqrt{f_{\text{дцп}}}}{1 + k_{\text{п/о}}^{\text{ц}}} =$$

$$= \frac{y\sqrt{1-k_{\text{д}}} + \sqrt{k_{\text{д}}} + 4\sqrt{0{,}1}}{1+4} = 0{,}2\,y\sqrt{1-k_{\text{д}}} + 0{,}2\sqrt{k_{\text{д}}} + 0{,}25. \qquad (3.55)$$

График полученной функции приведен на рис. 3.35.

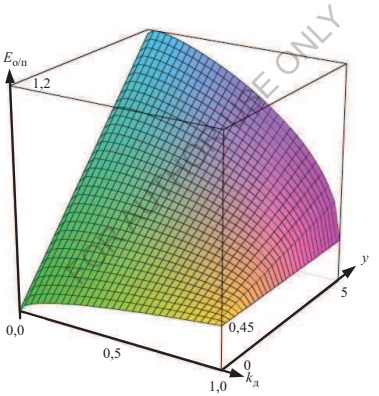


Рис. 3.35 Зависимость эффективности применения структур по критерию затрат на обработку и передачу данных при $k_{\text{п/о}}^{\text{ц}} = 4$, $f_{\text{дцп}} = 0{,}1$

При $E_{\text{о/п}} = 1$, $y = k_{21}\sqrt{n}$, $k_{\text{п/о}}^{\text{ц}} = 4$ получим следующую зависимость y от $k_{\text{д}}$ і $f_{\text{дцп}}$:

$$y = \frac{5 - \sqrt{k_{\text{д}}} - 4\sqrt{f_{\text{дцп}}}}{\sqrt{1-k_{\text{д}}}}. \qquad (3.56)$$

График плоскости, что разделяет области преимущества централизованных-децентрализованных структур приведены на рис. 3.36.

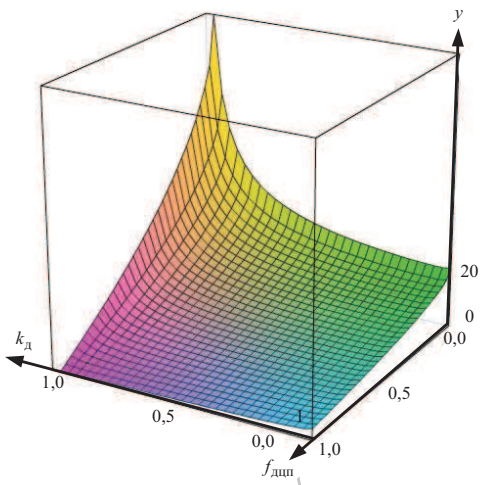


Рис. 3.36 Предельная плоскость между областями предпочтения централизованных-децентрализованных ЦУС

Область над плоскостью соответствует лучшим вариантам централизованных ЦУС, под плоскостью - децентрализованным системам. При снижении $f_{дцп}$ от 1 до 0 пределы децентрализованных систем расширяются. При незначительном уменьшении $f_{дцп}$ в области $k_д = 1$ диапазон для децентрализованных систем резко снижается. Таким образом, расширению области децентрализованных систем способствуют уменьшения y (или n), увеличение $k_{дцп}$ и снижение $f_{дцп}$.

Далее предлагаются модели сравнения структур ЦУС при условии суммарных затрат на обработку данных с диспетчеризацией.

В процессе выбора структуры ЦУС важно учитывать не только изменения в структурной организации систем, но и в алгоритмах их функционирования. Это связано в первую очередь с решением задач диспетчеризации и построением многоуровневой системы прерываний в централизованной системе, которая увеличивает ее применяемую вычислительную мощность. Будем ее учитывать с помощью коэффициента $f_{дисп}$, который назовем фактором диспетчеризации. Тогда

$$C^{ц}_{дисп} = k_1 (f_{дисп} W)^{g_1},\ f_{дисп} \geq 1. \qquad (3.57)$$

В соответствии с методикой, что применяется, вычисляем коэффициент отношения затрат (используя (3.50) і (3.57):

$$E_{\text{дисп}} = \frac{C_{\text{о}}^{\text{дц}}}{C_{\text{дисп}}^{\text{ц}}} = \frac{k_{21}\sqrt{n}\sqrt{1-k_{\text{д}}} + \sqrt{k_{\text{д}}}}{\sqrt{f_{\text{дисп}}}}. \qquad (3.58)$$

График зависимости (3.58) представлен на рис. 3.37. Предельная плоскость между областями преимущества централизованных-децентрализованных ЦУС, когда $E_{\text{дисп}} = 1$, описывается выражением

$$y = \frac{\sqrt{f_{\text{дисп}}} - \sqrt{k_{\text{д}}}}{\sqrt{1-k_{\text{д}}}}. \qquad (3.59)$$

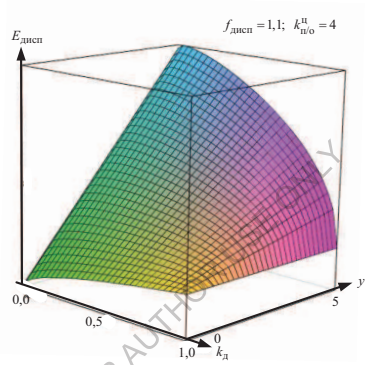


Рис. 3.37 Учет диспетчеризации при выборе структуры ЦУС

График функции (3.59) изображен рис. 3.38.

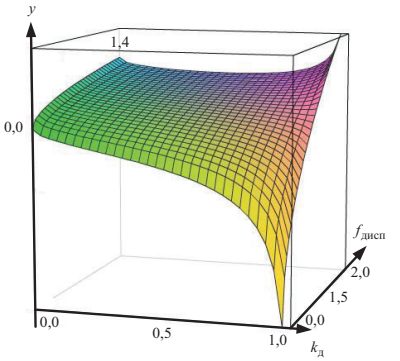


Рис. 3.38 Влияние расходов диспетчеризации
на распределение областей преимущества структур ЦУС

Приведенные графики показывают область преимущества децентрализованных структур за счет необходимости дополнительной мощности на организационные процедуры в центральной управляющей ЭВМ.

Модели сравнения структур ЦУС должны учитывать и расходы на резервирование подсистем. Известно, что для систем, работающих в реальном масштабе времени, предъявляются высокие требования к показателям надежности. К таким системам относятся и автоматизированные системы управления процессами расформирования-формирования поездов. Как правило, в системах используется горячее резервирование. Например, в системе АСУРСГ (станция Ясиноватая), построенной на базе мини-ЭВМ СМ-2М [69], центральная машина дублировалась с коммутацией сигналов на уровне устройств связи с объектом (УСО) (см. рис. 3.31). В децентрализованных системах на основе микропроцессорных промышленных ЭВМ обычно резервируют (дублируют) центральную, координирующую ЭВМ и для подсистем нижнего уровня выделяют один комплекс с возможностью его использования как скользящий резерв. На рисунке 3.32 резервные подсистемы показаны пунктирной линией. Любое резервирование связано с дополнительными ресурсными затратами, безусловно влияет на выбор соответствующей структуры.

Введем коэффициент степени резервирования $k_{\text{рез}}$, который принимает целые значения из области натуральных чисел $\mathbb{R}$, начиная с 2 ($k_{\text{рез}} \geq 2,\ k_{\text{рез}} \in \mathbb{R}$).

Тогда расходы для централизованных систем будут составлять

$$C^{\text{ц}}_{\text{рез}} = k_{\text{рез}} * C^{\text{ц}}_{\text{дисп}} = k_{\text{рез}} k_1 (f_{\text{дисп}} W)^{g_1}. \qquad (3.60)$$

При дублировании $k_{\text{рез}} = 2$. Будем предполагать, что для децентрализованной структуры используется тот же коэффициент, то есть дублируется центральная, координирующая подсистема и любая из n локальных подсистем специальной резервной подсистемой. В этом случае расходы на резервирование в децентрализованной системе можно оценить по формуле

$$C^{\text{дц}}_{\text{рез}} = k_{\text{рез}} * \left[k_2 ((1 - k_{\text{д}}) \frac{W}{n})^{g_1} + k_1 (k_{\text{д}} W)^{g_1} \right]. \qquad (3.61)$$

Эффективность организации резервирования по критерию минимальной стоимости будем определять коэффициентом

$$E_{\text{рез/о}} = \frac{C^{\text{дц}}_{\text{рез}}}{C^{\text{ц}}_{\text{рез}}} = \frac{k_{21}}{\sqrt{n}} * \sqrt{1-k_{\text{д}}} + \sqrt{k_{\text{д}}}\,. \qquad (3.62)$$

Выполним преобразования $\frac{k_{21}}{\sqrt{n}} = \frac{k_{21}\sqrt{n}}{n} = \frac{y}{n}$. Тогда плоскость, которая разделяет области эффективности, будет описываться выражением

$$\frac{y}{n} * \sqrt{1-k_{\text{д}}} + \sqrt{k_{\text{д}}} = 1. \qquad (3.63)$$

Откуда

$$y = \frac{n - \sqrt{k_{\text{д}}}}{\sqrt{1-k_{\text{д}}}}. \qquad (3.64)$$

На рис. 3.39 представлен график плоскости, разделяющей области преимущества резервирования в централизованных - децентрализованных системах.

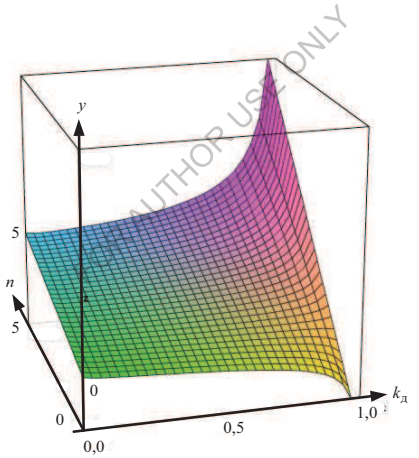


Рис. 3.39 Плоскость, что разделяет область преимущества централизованных (над плоскостью) и децентрализованных (под плоскостью) систем

Предложенные модели можно использовать отдельно при исследовании влияния отдельных характеристик системы на соответствующие потери и структуры. Но наиболее полезным может быть интегральный показатель эффективности структур ЦУС.

На основании описанных частных показателей сравнительной оценки централизованных-децентрализованных ЦУС (3.49), (3.50), (3.52), (3.53), (3.57),

(3.60), (3.61) предлагается комплексный или интегральный показатель, учитывающий процессы обработки и передачи данных, диспетчеризации заявок в централизованных системах и организации резервирования.

$$E_{\text{инт}} = \frac{C^{\text{дц}} + C^{\text{дц}}_{\text{п}} + C^{\text{дц}}_{\text{рез}}}{C^{\text{ц}}_{\text{дисп}} + C^{\text{ц}}_{\text{п}} + C^{\text{ц}}_{\text{рез}}}. \qquad (3.65)$$

Сделав подстановки и для уравнения плоскости $E_{\text{инт}} = 1$ получим следующее выражение

$$(1+k_{\text{д}})(\sqrt{k_{\text{д}}} - \sqrt{f_{\text{дисп}}}) + k^{\text{ц}}_{\text{п/о}}(\sqrt{f_{\text{дцп}}} - 1) + k_{21}\sqrt{1-k_{\text{д}}}(1 + \frac{k_{\text{рез}}}{\sqrt{n}}) = 0. \quad (3.66)$$

По (3.66) можно построить плоскость, разделяющую области преимущества централизованных-децентрализованных ЦУС, задав набор параметров системы. Например, если принять, что $f_{\text{дисп}} = 1{,}1$, $f_{\text{дцп}} = 0{,}9$, $k_{\text{рез}} = 2$, $k^{\text{ц}}_{\text{п/о}} = 1$ і $k_{21} = 1$, график будет иметь вид, представленный на рис. 3.40.

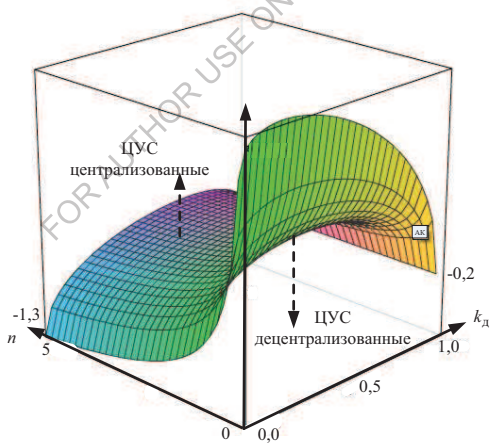


Рис. 3.40 Интегральная плоскость, что разделяет области преимущества централизованных и децентрализованных ЦУС

Полученная зависимость позволяет определить на ранних стадиях проектирования ЦУС при заданных параметрах функционирования область преимущества децентрализованных систем, которая увеличивается при снижении количества подсистем на нижнем ярусе n и уменьшении их мощности ($k_{\text{д}} \to 1$).

Предложенный метод и набор аналитических моделей позволяют на стадиях системотехнического проектирования и модернизации ЦУС обосновать выбор централизованной или иерархической (децентрализованной) структуры.

3.4.3 Подход к оценке затрат на построение ЦУС

В п. 3.4.1, 3.4.2 предлагаются две методики оценки и выбора вариантов построения распределенных цифровых управляющих систем (ЦУС) на примере автоматизированной системы управления расформированием-формированием составов на сортировочных горках. В этих работах недостаточно внимания уделяется учету расходов на средства обмена данными в исследуемых вариантах. В [65] эти расходы отнесены к расходам на микроконтроллерные подсистемы, а в [76] они задаются коэффициентом от затрат на устройства обработки данных и учитывают скорости передачи данных. В работе [144] приведены экспериментальные данные по расходам на обмен данными, приходится на централизованную систему и способ пересчета этих расходов при кольцевой сети. Хотя с точки зрения минимизации затрат на коммуникации на объекте оптимальный результат дает построение минимального остовного дерева. В данной части предлагается аналитическая модель для оценки затрат на обработку и обмен данными с минимальной суммарной длиной каналов передачи данных между подсистемами.

Основные предположения и исходные данные.

Предположим, что есть распределенная ЦУС, которая состоит из n локальных подсистем. Каждая ЭВМ обслуживает m точек автоматизации. Для сортировочных горок, в зависимости от путевого развития, количество таких точек колеблется от 321 до 850 [105] На рисунке 3.41 показаны централизованная система (а), распределенная-кольцевая (б) и система с минимальным скелетным деревом коммуникаций (в).

Будем считать, что общая длина соединительного кабеля (линии связи до m локальных точек автоматизации и магистральные линии связи), приведена к общей длине в централизованной системе, когда $n = 1$ (рис. 3.41 (а)). Тогда при заданном m и выбранном n можно получить $L_0 / L_1 = f(n,m)$.

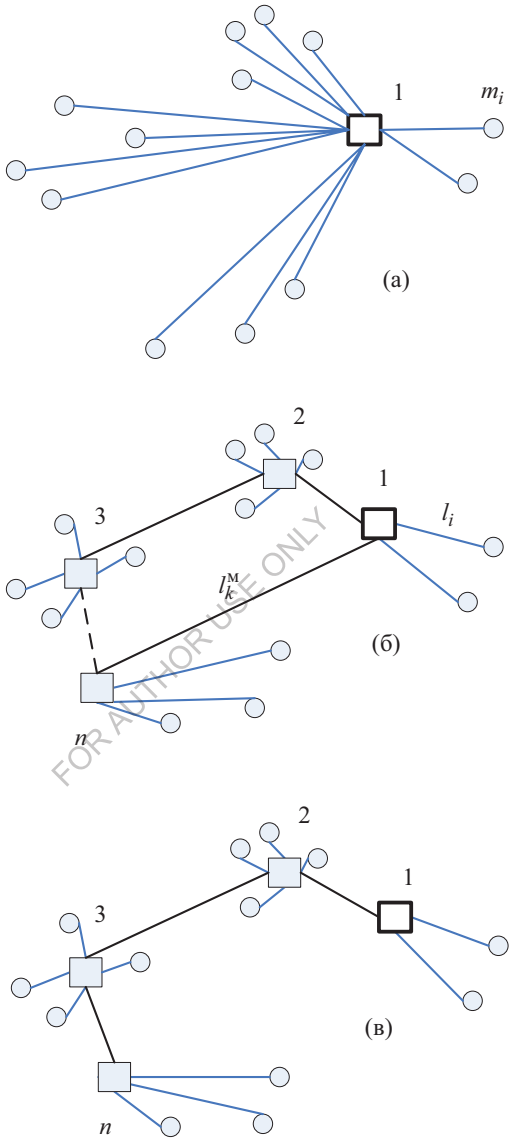


Рис. 3.41 Варианты структур ЦУС

Таблица 3.1

Значение функции $L_0/L_1 = f(n,m)$ **[144]**

	n						
	1	3	5	7	9	11	13
При $m = 144$ $L_0/L_1 = f(n,m)$	1,00	0,64	0,50	0,44	0,39	0,36	0,34
При $m = 3600$ $L_0/L_1 = f(n,m)$	1,00	0,62	0,43	0,34	0,30	0,27	0,26

Стоимость затрат на централизованную систему (n = 1) составляет

$$C^{\text{ц}} = C_1 = C_1^{\text{обр}} + C_1^{\text{каб}}. \qquad (3.67)$$

По экспериментальным данным установлено [144], что

$$C_1^{\text{обр}}/\text{C}_1 = 0,2 \quad \text{і} \quad C_1^{\text{каб}}/\text{C}_1 = 0,8. \qquad (3.68)$$

Будем считать, что при переходе от централизованной структуры к распределенной системы с n контурами управления, суммарная нужна вычислительная мощность W_0 равномерно распределяется между подсистемами, то есть распределенная ЦУС является однородной

$$W_0 = nW_i \quad \forall i \in \overline{1,n}. \qquad (3.69)$$

При этом

$$C_0^{\text{обр}} = aW_0^{g_1}, \text{ где } g_1 = 0,5. \qquad (3.70)$$

Определение длины кабеля в реальных системах зависит от множества факторов, поэтому точные расчеты практически невозможны. Координаты места точек автоматизации и размещения микроконтроллерных подсистем можно определить только после внедрения ЦКС. Тем более прокладки кабеля может вестись по построенным существующим и новым каналам. Для сортировочной горки можно считать, что распределение этих точек является равномерным.

На основании принятых предположений и (3.68), (3.69), (3.70) можно записать относительные затраты на микроконтроллеры в распределенной системе, приведеные в расходы в централизованной структуре

$$C_n^{\text{обр}} = n * C_i = n * a * \sqrt{\frac{W_0}{n}} = \frac{n * a * \sqrt{W_0}}{\sqrt{n}} = \sqrt{n} * C_1^{\text{обр}}. \qquad (3.71)$$

Взяв отношение, получим

$$\frac{C_n^{\text{обр}}}{C_1} = \frac{\sqrt{n} * C_1^{\text{обр}}}{C_1} = 0,2\sqrt{n}\,. \qquad (3.72)$$

Относительную стоимость кабельных соединений можно определить следующим образом:

$$\frac{C_n^{\text{каб}}}{C_1} = \frac{C_n^{\text{каб}}}{C_1^{\text{каб}}} * \frac{C_1^{\text{каб}}}{C_1} = \frac{C_n^{\text{каб}}}{C_1^{\text{каб}}} * 0,8 \approx \frac{L_0}{L_1} * 0,8 = 0,8 f(n,m)\,. \qquad (3.73)$$

Принимая (3.73), преобразуем таблицу 3.2 в таблицу 3.3.

Таблиця 3.2

Значение функции $0,8 f(n,m)$

	n						
	1	3	5	7	9	11	13
При $m = 144$ $L_0/L_1 = f(n,m)$	0,80	0,51	0,40	0,35	0,31	0,29	0,27
При $m = 3600$ $L_0/L_1 = f(n,m)$	0,80	0,50	0,34	0,27	0,24	0,22	0,21

Для решения задачи определения количества подсистем в распределенной структуре по критерию минимизации затрат на микроконтроллеры и линии связи можно предложить графический подход. На рис. 3.42 представлены графики соответствующих расходов $\frac{C_n^{\text{каб}}}{C_1}$; $\frac{C_n^{\text{обр}}}{C_1}$ і $\frac{C_n^{\text{каб}} + C_n^{\text{обр}}}{C_1}$.

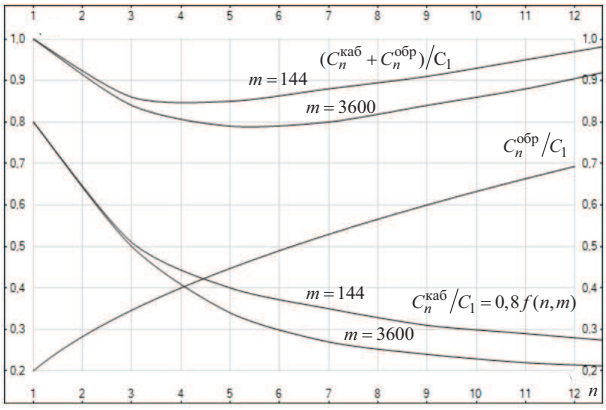


Рис. 3.42 Графические определения $\min\left\{(C_n^{\text{каб}} + C_n^{\text{обр}})/C_1\right\}$

Полученные результаты показывают, что минимум затрат соответствует 4-5 подсистемам (n = 4-5), то есть в иерархической структуре одна координирующая подсистема - на верхнем уровне, и на нижнем уровне - 3 или 4 технологические подсистемы. Эти результаты подтверждают полученные данные с помощью других аналитических моделей [65, 76], что подтверждает корректность построения моделей и возможность их практического применения в процессе проектирования новых и модернизации существующих ЦУС на сортировочных горках.

Полученные модели могут использоваться на стадии системотехнического проектирования цифровых управляющих систем после анализа объекта автоматизации, синтеза вариантов информационно-управляющей структуры (определение количества подсистем) и определения нужной суммарной производительности ЦУС (по методике проектирования систем автоматизации сортировочных горок [78]). В качестве расчетной структуры рекомендуется использовать не кольцевую структуру, а древовидную, представляющий минимальное скелетное дерево. В этом случае потребуются изменения в таблицах 3.1 и 3.2. Предложенные модели позволят повысить эффективность использования вычислительных и кабельных ресурсов ЦУС и сократить время их проектирования.

4. Развитие моделей и методов расчёта информационно-временных характеристик АСУ ркального времни

4.1 φ- транзакция как основная модель для оценки информационно-временных характеристик сервис-ориентированных систем

4.1.1 Определение понятия φ-транзакция

Для исследования и проектирования информационно-управляющих систем реального масштаба времени сортировочных станций введем понятие φ-транзакция (англ. Transaction, от лат. Transactio - соглашение, договор).

φ-транзакция - это логично объединенная последовательность функционально-алгоритмических и программных блоков операций (далее - ФПБ), которая обрабатывается полностью с момента появления в управляющей системе события, которая требует обработки, до момента завершения ее обработки с выдачей сообщения и / или управляющего воздействия на устройства объекта управления.

φ- транзакция должна обладать свойствами ACID [125]. Это аббревиатура четырех слов, означающих атомарность (atomicity), согласованность (consistency), изоляцию (isolation) и устойчивость (durability).

Атомарность. Атомарность представляет единицу работы. В отношении φ-транзакции это означает, что либо вся единица работы будет успешно выполнена, или ничего не будет изменено.

Согласованность. Состояния перед стартом φ- транзакции и после ее завершения должны быть корректными. Во время φ- транзакции состояние может иметь промежуточные значения.

Изоляция. Изоляция означает, что φ-транзакции, которые выполняются одновременно, изолированные от состояния, которое меняется во время φ - транзакции. φ-транзакция А не может видеть промежуточного состояния φ-транзакции В до тех пор, пока она не будет завершена.

Устойчивость. После завершения φ-транзакции ее результат должен быть зафиксирован на постоянной основе. Это означает, что если произойдет сбой микроконтроллера или управляющей ЭВМ, то состояние должено быть восстановленно после их перезапуска.

От известных определений понятия φ-транзакция отличается

расширенным описанием, необходимым для расчета характеристик АСУ на ранних стадиях ее проектирования, включающее для множества обрабатываемых системой φ-транзакций:

- описание используемых при выполнении φ-транзакций основных массивов базы данных $M\,(\forall m_i \in M\ \{O_i\}$, где O_i, $i \in \overline{1,|M|}$ - размер массива или объем памяти в байтах);

- описание множества событий E, которые требуют реакции системы управления, то есть запуска φ-транзакций $(\forall \varphi_j \in \Phi\ \{s_j, \lambda_j\}$, где $s_j \in S_o$ - множество входных сигналов и сообщений, связанных с соответствующими событиями E, λ_j - средняя интенсивность запуска j-й φ-транзакции);

- описание множества условий и требований успешного завершения φ-транзакций $\forall \varphi_j \in \Phi\ \forall d_k \in D\ \{op_{kj}, Q_{kj}, K_{kj}^{\text{г}}, t_{kj}^{\text{гр}}\}$, где d_{kj} - k-й приемник информации от j-й φ-транзакции, $op_{kj} = \{$ выдать сообщение $msg_{kj} \in Msg^{out}$, выдать управляющую действие в виде сигнала $s_{kj} \in S^{out}$ (2.24), ... $\}$, Q_{kj} - объем информации в байтах, передаваемого k-м приемнику от j-й φ-транзакции, коэффициент готовности и предельное время выполнения φ-транзакции j для k-го приемника;

- описание каждого ФПБ, который входит в состав системы: $\forall \phi_l \in ФПБ\ \forall m_i \in M\ \forall com_n \in \mathbb{C}\{k_{i,l}, op_{i,l}, K_{nl}\}$, где $ФПБ = \{\phi_l \,|\, l \in \overline{1,|ФПБ|}\}$ множество ФПБ системы, $M = \{m_i \,|\, i \in \overline{1,|M|}\}$ - множество массивов базы данных, $\mathbb{C} = \{com_n \mid n = \overline{1,|\mathbb{C}|}\}$ - множество типов команд в смеси команд проектируемой системы или микс (mix), $0 \le k_{i,l} \le 1$ - коэффициент, показывающий какая часть массива i используется при одном исполнении ФПБ l); $op_{j,l,i} = \{$чт,зп$\}$ - операции с памятью i при выполнении l-го ФПБ в j-й φ-транзакции, K_{nl} - количество команд типа n в l-м ФПБ.

Общий вид φ-транзакций и описание системы, которая проектируется показаны на рис. 4.1.

В наборе ФПБ, входящих в φ-транзакцию, выделяются вершины, в которых возможно множественное ветвление процесса обработки данных (например, ϕ3 имеет три варианта продолжения вычислений). В этом случае

задается вероятность выбора направления ветвления, сумма которых равна 1 (в примере, $p_{35} + p_{37} + p_{32} = 1$).

Все описанные характеристики φ-транзакций используются в процессе проектирования систем автоматизации сортировочных станций в соответствии с предложенным подхода КСИ (рис. 2.15, п. 2.4.4). Фактически множество φ-транзакций (Ф) является М-модель системы, в которой исключаются ФПБ, которые повторяются. φ-транзакции "расшивают", структурируют М-модель по отдельным сервисам (далее - заявкам), для которых специфицированы определенные показатели (условия) качества предоставления сервиса.

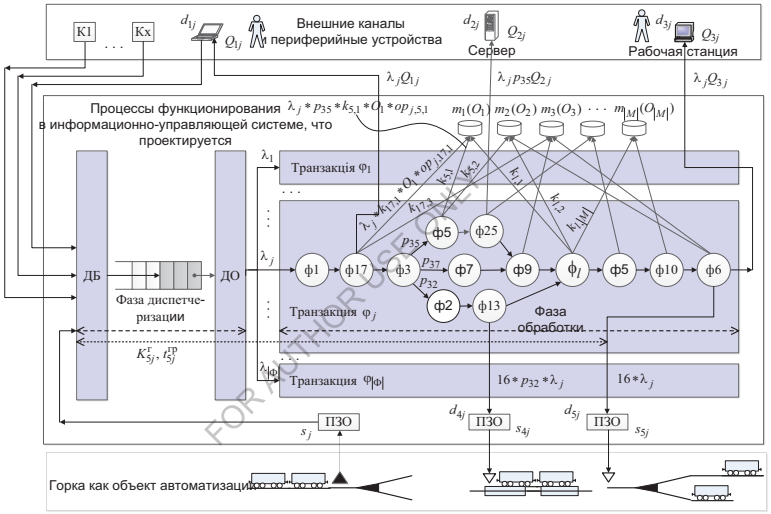


Рис. 4.1 φ-транзакции в системах автоматизации сортировочных станций

4.1.2 Метод оценки характеристик ЦУС реального времени в процессе проектирования

Исходными данными для расчета характеристик проектируемой системы управления являются:

- описание функциональных программных блоков, который включает список ФПБ, используемые ими массивы и коэффициенты их использования при каждом выполнении ФПБ, количество операций выбранных типов в каждом ФПБ (см. таблицу 4.1);

- описание φ-транзакций в виде ориентированных вершинных и реберно-

взвешенных графов с описанием их характеристик (см. рис. 4.1);

- описание выбранного заранее типа процессора и времена выполнения операций упомянутых выше типов (примеры описания микропроцессоров - в таблицах 4.2, 4.3 [139]).

Таблица 4.1

Опис функціональних програмних блоків

Параметры	Функциональные программные блоки					
	ϕ_1	ϕ_2	...	ϕ_l	...	$\phi_{\lvert ФПБ \rvert}$
Смесь команд (микс) системы	Количество операций соответствующего типа в ФПБ, тыс. команд					
com_1	K_{11}	K_{12}		K_{1l}		$K_{1\lvert ФПБ \rvert}$
com_2	K_{21}	K_{22}		K_{2l}		$K_{2\lvert ФПБ \rvert}$
...						
com_n	K_{n1}	K_{n2}		K_{nl}		$K_{n\lvert ФПБ \rvert}$
...						
$com_{\lvert \mathbb{C} \rvert}$	$K_{\lvert \mathbb{C} \rvert 1}$	$K_{\lvert \mathbb{C} \rvert 2}$		$K_{\lvert \mathbb{C} \rvert l}$		$K_{\lvert \mathbb{C} \rvert \lvert ФПБ \rvert}$
Массивы и их объемы	Коэффициент использования массива и тип операции					
$m_1(O_1)$	$k_{1,1}$, $op_{1,1}$	$k_{1,2}$, $op_{1,2}$		$k_{1,l}$, $op_{1,l}$		$k_{1,\lvert ФПБ \rvert}$, $op_{1,\lvert ФПБ \rvert}$
$m_2(O_2)$	$k_{2,1}$, $op_{2,1}$	$k_{2,2}$, $op_{2,2}$		$k_{2,l}$, $op_{2,l}$		$k_{2,\lvert ФПБ \rvert}$, $op_{2,\lvert ФПБ \rvert}$
...						
$m_i(O_i)$	$k_{i,1}$, $op_{i,1}$	$k_{i,2}$, $op_{i,2}$		$k_{i,l}$, $op_{i,l}$		$k_{i,\lvert ФПБ \rvert}$, $op_{i,\lvert ФПБ \rvert}$
...						
$m_{\lvert M \rvert}(O_{\lvert M \rvert})$	$k_{\lvert M \rvert,1}$, $op_{\lvert M \rvert,1}$	$k_{\lvert M \rvert,2}$, $op_{\lvert M \rvert,2}$		$k_{\lvert M \rvert,l}$, $op_{\lvert M \rvert,l}$		$k_{\lvert M \rvert,\lvert ФПБ \rvert}$, $op_{\lvert M \rvert,\lvert ФПБ \rvert}$

Все исходные данные представляются в табличном виде для дальнейшей обработки.

Следует отметить, что ФПБ является или программами (если данный блок взят из библиотеки программ разработчика), или алгоритмами (для известных

алгоритмов управления горкой), или функциями (для новых задач). Это определяет точность оценки K_{nl} и $k_{i,l}, op_{i,l}$.

Таблица 4.2

Время выполнения арифметических операций, нс

Тип аргумента	int				double		
Операция	^	+	*	/	+	*	/
Pentium 4 2,5 ГГц	0,78	0,86	5,6	19	2	2,8	15
CoreDuo 1,8 ГГц	1,1	1,1	2,2	4,2	1,7	2,7	18
Pentium M 1,8 ГГц	1,1	1,1	2,2	8,7	1,7	2,7	17
Athlon 64X2 2,5 ГГц	1,4	1,4	1,6	17	1,8	1,8	8,3

Таблиця 4.3

Число тактов, необходимых для выполнения инструкций на процессорах Intel P4+

Инструкция	XOR	ADD	MUL	DIV	FADD	FMUL	FDIV (double)
Задержка обработки	0,5-1	0,5-1	3-18	22-70	5-6	7-8	38-40
Время обработки	0,5	0,5	1-5	23-30	1	2	38-40

Для выбора набора команд $\mathbb{C}$ и получения K_{nl} можно воспользоваться методикой "ПОЭТ" [105]. Для новых функций существуют характеристики типовых алгоритмов, которые рекомендуются для стадий системного проектирования ИУВМ (див. табл. 4.4) [37].

Для расчета быстродействия вычислительных систем с целью их сравнения и выбора используется большое количество разнообразных моделей, методов и бенчмарков (benchmark - эталонные тесты Linpack, SPEC, SPECfp95, SPECweb97, TPC, WinMark, Winstone 97 и т.д.), но большинство из них ориентированы на уже разработанные системы и их практически невозможно использовать на ранних стадиях проектирования [37, 121, 148].

В разработанной методике CoDeCS предлагаются два способа оценки быстродействия процессоров - средняя быстродействие *i*-й модели на смеси задач системы $\tilde{W}_i$ и средняя быстродействие *i*-й модели на смеси φ - транзакций системы управления, что проектируется, $\tilde{W}_i^{\varphi}$.

Таблица 4.4

Характеристика типових алгоритмів

Класс алгоритмов (задач)	Входные параметры алгоритмов	Результативный характеристики		
		Внутренняя связность	Число средних операций	Удельный объемность
Первичная обработка измерений масштабирование)	n - число измерений, m - размер таблицы градуировки	$n+m$	nm	m
Статистический анализ	n - число измерений, m - число факторов, число коэффициентов	nm	nm^2	n
Оптимизация (линейное прил. И др.)	m - число переменных, n - число ограничений	$n(m+n)$	$n^2(m+n)$	n
Расчет технико-экономических характеристик	n - число данных, m - число констант формулы	$n+m$	nm	m

Для автоматизированных систем управления сортировочной горкой автором уже были определены частотные смеси команд, впервые описаны в работе [83]. Однако, очевидно, что их применение целесообразно только в том случае, если мы ориентируемся на набор задач системы АСУРСГ. Для учета новых алгоритмов и программ управления нужно корректировки частотных характеристик смесей. В качестве $\mathbb{C}$ рекомендуется использовать наборы команд из таблиц 4.2 и 4.3 или методику экспериментального программирования "ПОЭТ".

Быстродействие на частотной смеси команд при проектировании системы для заранее выбранного i-го типа МКП вычисляется следующим образом:

$$\tilde{W}_i = \frac{1}{\sum_{n=1}^{|\mathbb{C}|} \xi_n \tau_{ni}}, \qquad (4.1)$$

где ξ_n - частота или вероятность появления операции n в процессе

функционирования системы; τ_{ni} - среднее время выполнения команды n на микропроцессоре i-го типа; единицей измерения быстродействия является количество операций на смеси задач системы в секунду - оп/с.

Значение ξ_n вычисляется как

$$\xi_n = \frac{K_n}{\sum_{h=1}^{|\mathbb{C}|} K_h}. \qquad (4.2)$$

Тут K_h - количество команд h-го типа, которые обрабатываются системой в процессе выполнения φ-транзакций.

K_h вычисляется по формуле (4.3) с учетом данных K_{nl} для каждого ФПБ из таблицы 4.1 и структуры всех φ-транзакций, обрабатываемых системой, с учетом интенсивностей их запуска, описанных на рис.4.1.

$$\forall h \in |\mathbb{C}| \quad K_h = \sum_{\forall \varphi_j \in \Phi} \lambda_j * \sum_{\forall \phi_l \in \varphi_j} k_{lj} * K_{hl}. \qquad (4.3)$$

В формуле (4.3) $0 < k_{lj} \le 1$ - коэффициент, учитывающий снижение количества операций в φ-транзакции j в ФПБ l за счет предыдущих разветвлений и циклов (во взвешенном графе). Этот коэффициент исчисляется в процессе преобразования структуры и исчислении характеристик φ-транзакций (см. следующий п. 4.2).

Полученное среднее быстродействие i-й модели на смеси задач будущей системы $\tilde{W}_i$ используется в предложенной методике КСИ для сравнения различных типов МКП при их выборе, а также для ресурсосберегающих методов поиска рациональных структур децентрализованных систем управления сортировочной горкой (см. п. 3.4.2, 3.4.3).

Для сервис-ориентированных систем реального масштаба времени, в которых установлено ограничение на время реакции или время обслуживания каждой φ-транзакции, предлагается вычислять среднюю быстродействие i-й модели на смеси φ-транзакций, или $\tilde{W}_i^{\varphi}$, которое измеряется в количестве обрабатываемых средневзвешенных φ-транзакций в секунду ("$\varphi-$tps" - φ-transaction per second) или в минуту - "$\varphi-$tpm". Для этого воспользуемся выражением (4.4).

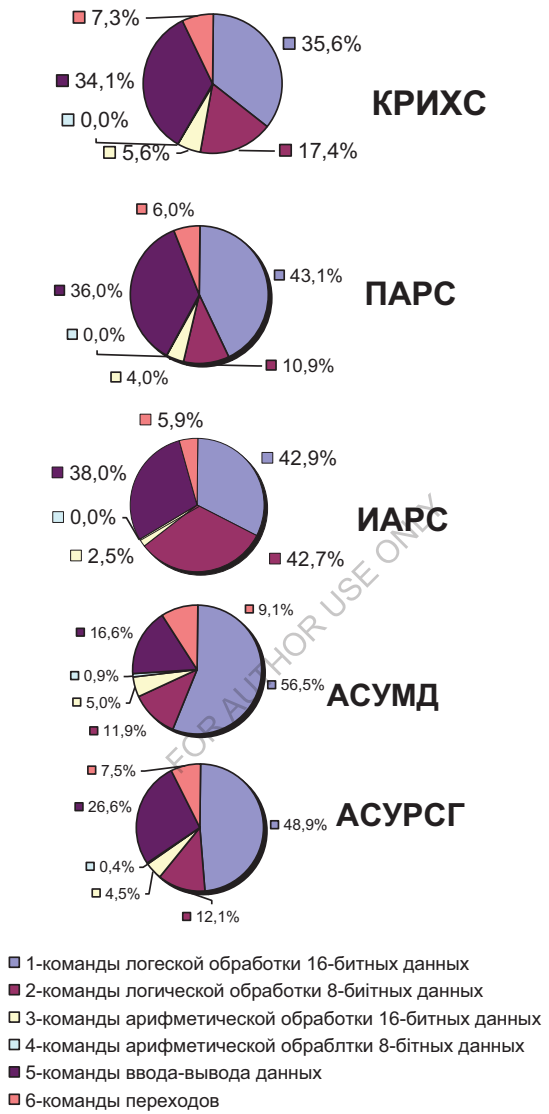


Рис. 4.2 Частотные смеси операций основных функциональных подсистем и системы управления сортировочной горкой

$$\tilde{W}_i^{\varphi} = \frac{\sum_{j=1}^{|\Phi|} \lambda_j}{\sum_{j=1}^{|\Phi|} \lambda_j T_{ji}}. \qquad (4.4)$$

В данной формуле, поскольку должно выполняться условие для выбранного *i*-го типа МКП (или тактовой частоты)

$$\forall j \in |\Phi| \;\; T_{ji} \le t_j^{\text{гр}}, \qquad (4.5)$$

то за (4.4) можно провести выбор для проекта такого типа МКП минимальной производительности (и стоимости), при котором выполняется (4.5). Метод вычисления T_{ji} изложен в следующем пункте.

Знание разработчиком в процессе проектирования системы управления реального времени значений $\forall j \in |\Phi| \;\; T_{ji}$ позволяет для выбранного *i*-го типа микропроцессора оценить, какую долю загрузки процессора дает предоставления *j*-го сервиса (*j*-й φ-транзакции)

$$\rho_{ji} = \lambda_j T_{ji} \quad \text{i} \quad \forall j \in |\Phi| \;\; \rho_{ji} \le 0{,}6. \qquad (4.6)$$

Выполняя распределение обрабатываемых φ-транзакций по полученным в результате структурной оптимизации подсистемах (см. описание КСИ), необходимо следить, чтобы в проекте общая загрузка сервисами каждой подсистемы $f_p \in \mathbb{F}$ не превышала допустимого предела (для процессоров это 0,6), т.е.

$$\forall f_{pi} \in \mathbb{F}_i \;\; R_{pi} \le 0{,}6. \qquad (4.7)$$

Иногда устанавливают и нижний предел загрузки (4.8) с целью эффективного использования вычислительных ресурсов и снижение общей стоимости ИУВМ.

$$\forall f_{pi} \in \mathbb{F}_i \;\; 0{,}1 \le R_{pi} \le 0{,}6. \qquad (4.8)$$

Построение φ-транзакций позволяет оценить ряд других важных для проектирования АСУ характеристик, в частности тех, которые связаны с информационными потоками в будущей системе. В АСУ будем выделять три типа информационных потоков: потоки сообщений (msg), потоки управляющих воздействий (сигналов, s_k) и потоки на чтение и запись информации из/в массивы будущей базы данных.

Информационный поток сообщений, формируется каждой φ-транзакцией

j, и передается на периферийное устройство d_k

$$\pi_{jd_k}^{msg} = \sum_{\forall \phi_l \in \varphi_j \wedge \exists d_k} \lambda_j \bar{p}_{jl} Q_{ld_k j} . \qquad (4.9)$$

Общий суммарный поток сообщений на приемник d_k в процессе работы системы будет составлять

$$\Pi_{d_k}^{msg} = \sum_{j=1}^{|\Phi|} \pi_{jd_k}^{msg} . \qquad (4.10)$$

У φ - транзакциях могут формироваться потоки управляющих воздействий, или сигналов s_k, которые выводятся через УСО на устройстве объекта управления. Для одной j-й транзакции и при кодировании одного действия 2-мя байтами (16 бит) они составят

$$\pi_{s_{kj}} = \sum_{\forall \phi_l \in \varphi_j \wedge \exists s_k} 16\lambda_j \hat{p}_{jl} . \qquad (4.11)$$

В системе в целом поток сигналов s_k будет составлять

$$\Pi_{s_k} = \sum_{j=1}^{|\Phi|} \pi_{s_{kj}} . \qquad (4.12)$$

Потоки информации, связанные с чтением / записью данных из / в массив i, при выполнении j-й транзакции можно вычислить таким образом:

$$\pi_{ji}^{\text{чт/зп}} = \sum_{\forall \phi_l \in \varphi_j} \lambda_j \bar{p}_{jl} k_{j,l,i} O_i op_{j,l,i} . \qquad (4.13)$$

Суммарный информационный поток в массив i в процессе работы системы можно вычислить по формуле

$$\Pi_i^{\text{чт/зп}} = \sum_{j=1}^{|\Phi|} \pi_{ji}^{\text{чт/зп}} . \qquad (4.14)$$

Общий информационный поток ко всем массивам или поток в центральную базу данных АСУ будет составлять

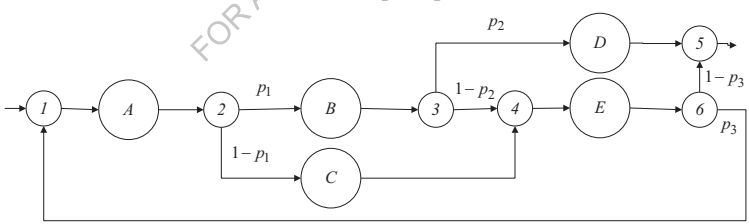
$$\Pi_M^{\text{чт/зп}} = \sum_{i=1}^{|M|} \Pi_i^{\text{чт/зп}} . \qquad (4.15)$$

4.2 Метод обработки φ - транзакций и расчет их характеристик

φ-транзакции, определенные в п. 4.1, фактически графами со взвешенными вершинами и ребрами (рис. 4.1). Для обработки таких графов существует большое разнообразие методов и методик, которые зависят от поставленных задач [121, 26]. Здесь предлагается метод обработки, ориентированный на автоматизацию процессов преобразования φ-транзакций и оценку их характеристик, используемых в процессе проектирования ИУВМ.

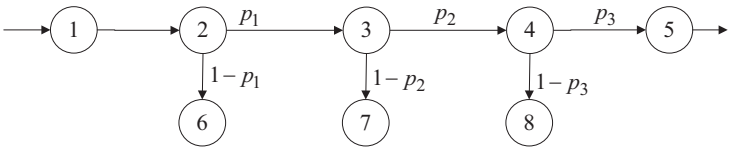
4.2.1 Метод оценки математического ожидания времени выполнения φ-транзакций

Преобразуем схему φ-транзакции в соответствии с приведенным выше правилам [26]. Исключим ложные объединения и включим те объединения, которые не показаны явно. Заменяем элементы принятия решения эквивалентными решениями, в которых все работы (время) включаются в участки, предшествующие решению или идут за ними. Объединяем элементы отделки, которые выполняются последовательно, заменяя их одной вершиной, которая эквивалентным образом представляет последовательность вершин, что заменены. Новая схема может иметь, например, такой вид:



Каждая вершина (объединения или принятия решения) задается индивидуальной меткой. Вероятность ставится в соответствие каждому выходу всех элементов принятия решения.

Вероятности путей. Предположим, что в φ-транзакции отсутствуют циклы. Рассмотрим следующий пример, не обращая внимания сейчас на величину работы в каждой вершине:

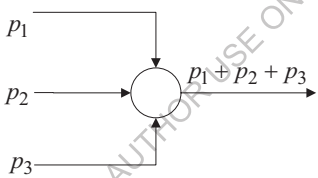


Сумма вероятностей по всем выходам должна равняться 1. Вероятность пребывания в некоторой вершине равна произведению вероятностей для путей, ведущих к вершине. Таким образом, вероятности, соответствующие различным путям, составляют

$$p_{12}=1,\ p_{23}=p_1,\ p_{14}=p_1p_2,\ p_{15}=p_1p_2p_3,$$
$$p_{16}=1-p_1,\ p_{17}=p_1(1-p_2),\ p_{18}=p_1p_2(1-p_3),$$

Аналогичным образом мы можем вычислить вероятности, подходящие другим путям, например (3, 8). Она представляет собой произведение вероятностей отдельных ребер *i*, следовательно, равна $p_2(1-p_3)$.

Вероятность, соответствующая выхода элемента объединения, то есть сумма вероятностей, соответствующих входам этого элемента:



Применим эти правила к нескольким примерам. В каждом случае мы определим вероятность расположения ребра вдоль пути, то есть вероятность того, что мы пройдем это ребро, начав движение с определенного входа.

Рассмотрим методику оценки временных параметров φ-транзакций.

Оценка временных параметров схем φ-транзакций выполняется в следующем порядке:

1. Упрощают блок-схему путем удаления избыточных объединений, слияние последовательных обработок и т. Д. Наносят на блок-схему вероятности для отдельных выходов элементов принятия решения. По мере необходимости заменяют элементы принятия решения эквивалентными элементами. Наносят оценки времени обработок на ребра блок-схемы.

2. Вычисляют вероятность прохождения каждого ребра.

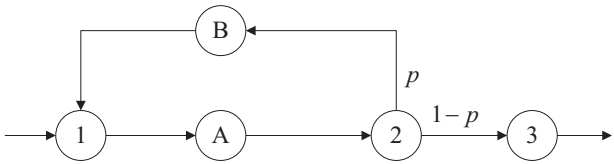
3. Умножают вероятность каждого ребра на соответствующее время. Таким образом, получают ожидаемое время для каждого ребра.

Для каждого возможного пути от входов к данному выходу получают

ожидаемое время прохождения в виде суммы всех ожидаемых времен ребер, встреченных на этом пути.

Наиболее сложными для расчетов является φ-транзакции с циклами.

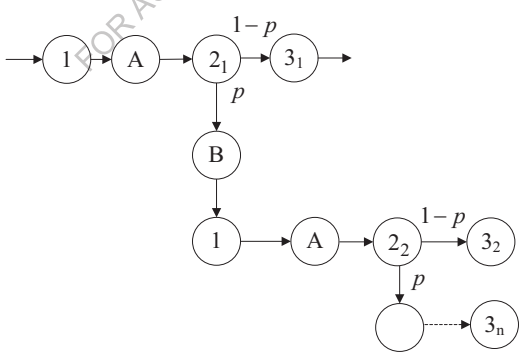
Одиночные циклы. Рассмотрим следующую схему с циклом:



Начинаем процесс в блоке 1, выполняем блок А и идем к принятию решения 2. Здесь с вероятностью происходит возврат в цикл и выполнен ние блока. Существует вероятность того, что вычислительный процесс выйдет из цикла. При возвращении в цикл работа повторяется по прохождению через блок 1.

Нас интересует вопрос, как анализировать работу циклических программ. Попробуем свести этот случай к уже изученной ситуации - программы с одним входом и многими выходами. Для того чтобы это сделать, мы повторим на схеме циклическую часть столько раз, сколько выполняться цикл.

В результате получаем



Такую модифицированную схему можно расширять до бесконечности. Для каждого выхода мы можем подсчитать вероятность точно так же, как мы это делали для графа с одним входом и многими выходами. Эти вероятности равны: для первого выхода $(1-p_1)$, для второго выхода $p(1-p_1)$, для третьего

выхода $p^2(1-p_1)$, ..., для n-го выхода $p^{n-1}(1-p_1)$.

Суммируя эти вероятности, получим

$$p_{общ} = \sum_{m=0}^{\infty}(1-p)p^n = (1-p)\sum_{m=0}^{\infty}p^n. \qquad (4.16)$$

Однако, поскольку

$$\sum_{n=0}^{\infty}p^n = \frac{1}{1-p} \text{ для } p<1, \qquad (4.17)$$

то, как и следовало ожидать,

$$p_{общ} = (1-p)\frac{1}{1-p} = 1.$$

Теперь мы можем вывести формулу уравнения цикла для работы в простом цикле. Вероятность n-кратного выполнения цикла равна

$$p^n(1-p).$$

Временные затраты, связанные с n-кратным прохождением по циклу, составляют

$$A + n(A+B).$$

Таким образом, ожидаемое время равно

$$T = \sum_{n=0}^{\infty}p^n(1-p)[A+n(A+B)], \qquad (4.18)$$

или в другом виде

$$T = (1-p)A\sum_{n=0}^{\infty}p^n(1-p)A + n(A+B)\sum_{n=0}^{\infty}np^n. \qquad (4.19)$$

Но, поскольку

$$\sum_{n=0}^{\infty}p^n = \frac{1}{1-p} \text{ і } \sum_{n=0}^{\infty}np^n = \frac{1}{(1-p)^2} \text{ для } p<1, \qquad (4.20)$$

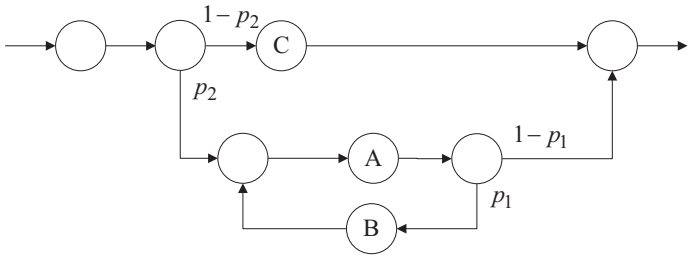
то из этого следует

$$T = \frac{(1-p)A}{(1-p)} + \frac{p(1-p)(A+B)}{(1-p)^2}, \qquad (4.21)$$

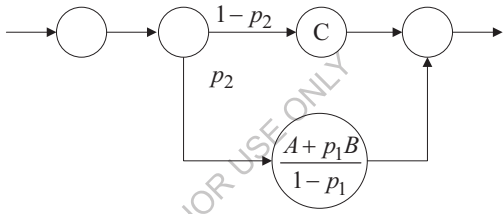
$$t = A + \frac{p(A+B)}{1-p}, \qquad (4.22)$$

$$T = \frac{A - Ap + Ap + Bp}{1-p} = \frac{A+Bp}{(1-p)}. \qquad (4.23)$$

Здесь используется теория бесконечных рядов и правила их суммирования. Итак, получена формула, позволяющая работать с циклами. Где бы в схеме φ-транзакции не встречался цикл, теперь можно заменить его эквивалентным нецикличным процессом. Итак,

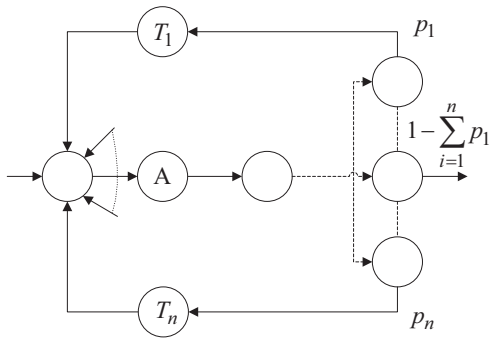


можно заменить на

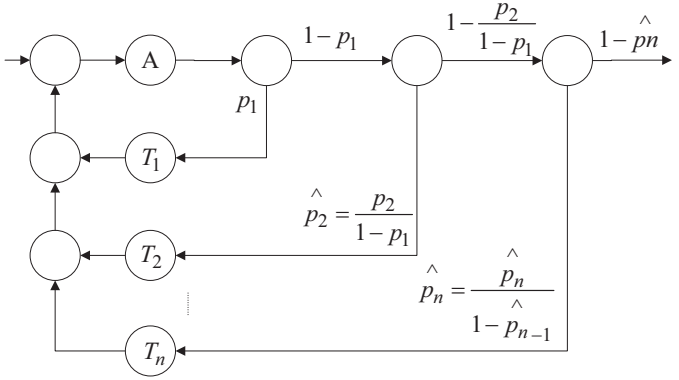


Чтобы упростить алгебраические выкладки, в дальнейшем будем обозначать вероятность для ребра, что образует цикл, через p_1. На практике будем начинать с внутреннего цикла и двигаться в сторону выхода.

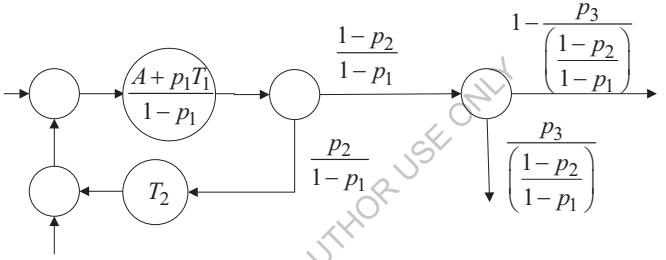
Рассмотрим более сложные циклы. Есть разветвленный процесс со многими выходами и с петлей обратной связи, подобный



Он может быть заменен следующей эквивалентной блок-схеме:



Исключая внутренний цикл, мы получим



В этом случае для схемы с многими циклами может быть получена следующая формула:

$$T_n = \frac{A+\sum_{i=1}^{n} p_i T_i}{1-\sum_{i=1}^{n} p_i} \ . \qquad (4.24)$$

4.2.2 Автоматизация построения и расчета характеристик φ-транзакций

Разработанные графовые и аналитические средства были положены в основу программы CSProject для автоматизированного формирования транзакций системы, которые привязаны к сигналам участков технологических процессов сортировочных станций. Исходные данные для расчетов описанные

в п. 4.1. Они могут вводиться в программу двумя способами: с предварительно подготовленных таблиц Excel специальной структуры, фрагмент которой представлен на рисунке 4.3; в интерактивном режиме с помощью специально разработанного графического редактора транзакций и их параметров (см. рис. 4.4). Фрагмент результатов, полученных с помощью CSProject приведен на рис. 4.5.

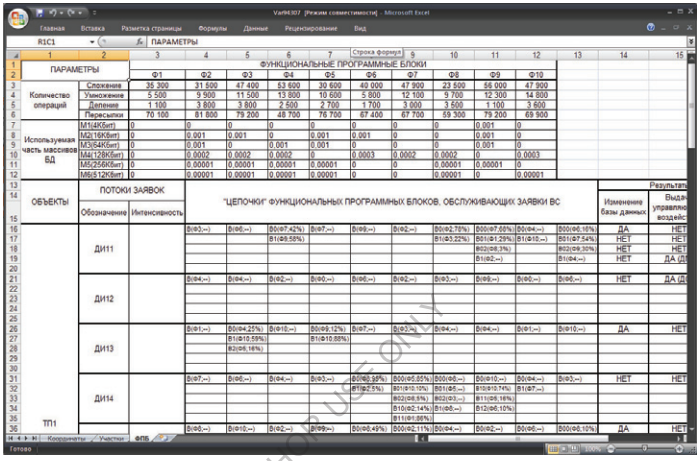


Рис. 4.3 Табличная формализация исходных данных для проектирования

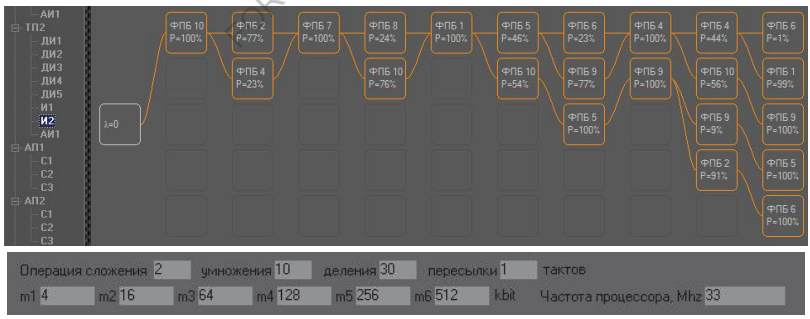


Рис. 4.4 Окно редактора ввода транзакции для сигнала И2 из технологического процесса (ТП2) и окно задания параметров массивов и МКК

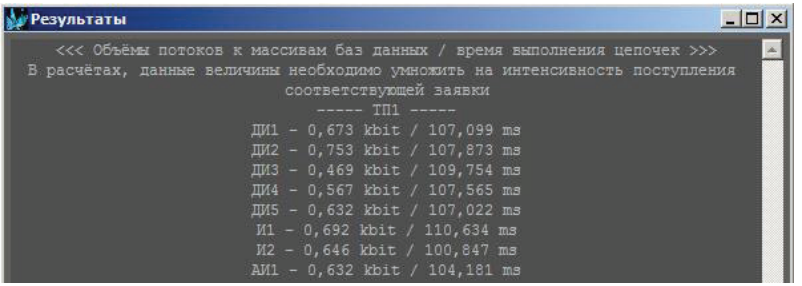


Рис. 4.5 Окно результатов обработки φ-транзакции

4.3 Методика, аналитические модели и инструментальные средства выбора системы приоритетов обработки φ-транзакций

Анализ работы сервис-ориентированных систем реального времени управления сортировочными станциями в виде набора φ-транзакций обрабатываемых с жесткими временными ограничениями показывает, что в условиях предыдущего выбранного типа процессора i должны выполняться множество условий (4.25).

$$\forall j \in |\Phi| \ \forall d_k \in D \ \ t(\varphi_{ijd_k}) = t_{ijd_k} = T_{ijd_k}(\phi.\partial) + T_{ijd_k}(\phi.o) = \\ = \tau_{wijd_k} + T_{ijd_k} \le t_{ijd_k}^{\text{гр}}. \qquad (4.25)$$

То есть для каждой транзакции j и каждой точки d_k вывода сообщений или управляющих воздействий этой транзакцией, сумма времен выполнения фазы диспетчирования (ф.д) и фазы обработки транзакции (ф.о.) не должно превышать установленных предельных значений $t_{ijd_k}^{\text{гр}}$.

Поскольку T_{ijd_k} зависит только от выбранного типа процессора и, в данной постановке, является фиксированным, то условие (4.25) можно представить в виде

$$\forall j \in |\Phi| \ \forall d_k \in D \ \ \tau_{wijd_k} \le t_{ijd_k}^{\text{гр}} - T_{ijd_k}, \qquad (4.26)$$

τ_{wijd_k} можно варьировать, выбирая дисциплину обслуживания заявок, поступающих в систему.

Продолжительность интервалов времени между поступлением заявок в ЦУС случайно. В связи с этим могут возникнуть очереди заявок, ожидающих обслуживания процессором. Организацию очередей осуществляют

специальные программы, которые называются «Диспетчерами». Схема обслуживания заявок в ЦУС показана на рисунке 4.6.

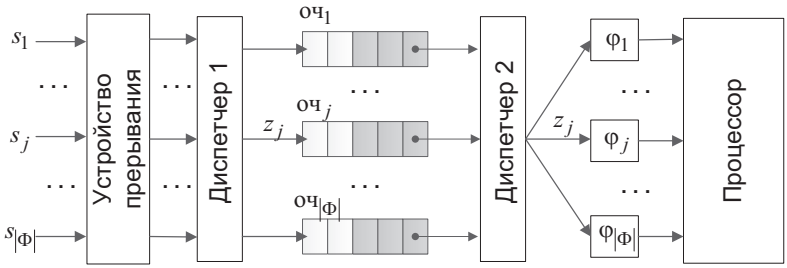


Рис. 4.6. Схема обслуговування заявок в ЦУС

После поступления сигнала s_j прерывается работа процессора, возможно обслуживает заявку z_i, выполняя транзакцию φ_i. Устройство прерывания инициирует работу программы «Диспетчер 1», которая ставит заявку z_j в общем случае в одну из очередей $оч_1,...,оч_{|\Phi|}$. Далее процессор продолжает прерванную работу. После ее завершения управление передается программе «Диспетчер 2», которая осуществляет анализ очередей для выбора заявки на обслуживание. В соответствии с кодом выбранной заявки начинается решение φ-транзакции. Если очереди пустые, то процессор находится в состоянии ожидания или решает так называемые «фоновые» задачи, которые не требуют срочности.

Правила, по которым осуществляется выбор заявок из очередей, называются дисциплиной обслуживания. Если определенным типам заявок не предоставляется преимущество (приоритет), то дисциплину обслуживания называют безприоритетною. В этом случае используется одна очередь заявок. Выбор заявок z_j на обслуживание, как правило, выполняется с начала очереди или с ее конца. Первую дисциплину называют дисциплиной FIFO (First Input - First Output), а вторую - дисциплиной LIFO (Last Input - First Output).

Названные дисциплины обеспечивают одинаковое среднее время ожидания заявок в очереди (τ_{wi}),однако дисциплина FIFO минимизирует дисперсию τ_{wi} и в связи с этим применяется чаще всего.

Характеристика распределения вероятности возникновения интервалов

между заявками определяется особенностями объекта управления. В данном случае в задачах автоматизации сортировочных станций принимаем гипотезу про экспоненциальный характер этого распределения [138].

Поток заявок с экспоненциальным распределением вероятности называется простым, если выполняются условия независимости поступления заявок и невозможности одновременного поступления нескольких заявок. Простые потоки при объединении образуют также простой поток, причем их интенсивности λ_j суммируются. Для простого потока характерно преобладание коротких интервалов между заявками. Поэтому создается жесткий режим работы системы. Оценки качества функционирования ИУВМ, полученные в этом случае, являются предельными.

Продолжительность обслуживания τ_{si} при наличии различных по трудоемкости ветвей программ может быть случайной. Ее среднее значение определяется по методике, изложенной в п. 4.2.1.

Коэффициент вариации ν_{si} величины τ_{si} может меняться от 0 при постоянном времени обслуживания до 1 при экспоненциальном распределении τ_{si}.

К важным характеристикам качества ЦУС относят загрузки системы ρ_i, среднее время ожидания τ_{wi} и среднее время пребывания заявки в системе (время ответа) τ_{sysi}. Размер загрузки системы j-м потоком заявок для i-го процессора $\rho_{ji} = \lambda_j \tau_{sji}$. Полная загрузка от всех потоков $R_{|\Phi|i} = \sum_{j=1}^{|\Phi|} \rho_{ji}$.

Очевидно, что среднее время ответа можно найти по формуле:

$$\tau_{sysji} = \tau_{sji} + \tau_{wji} \ . \qquad (4.27)$$

Для безприоритетных дисциплин обслуживания (БП) среднее время ожидания для всех типов заявок одинаков и может быть рассчитан по формуле:

$$\tau_{wi} = \frac{\sum_{j=1}^{|\Phi|} \rho_{ji} \cdot \tau_{sji} \cdot (1 + \nu_{sji}^2)}{2 \cdot (1 - R_{|\Phi|i})}, \qquad (4.28)$$

где $|\Phi|$ – количество типов заявок (количество φ-транзакций).

Из формулы (4.28) видно, что τ_{wi} увеличивается по мере роста ν_{sji}. Так, при экспоненциальном законе распределения τ_{sji} вдвое больше, чем при постоянном времени обслуживания. Это позволяет упростить (4.28).

$$\tau_{wi} = \frac{\sum_{j=1}^{|\Phi|} \rho_{ji} \cdot \tau_{sji}}{(1 - R_{|\Phi|i})}. \qquad (4.29)$$

Отличие срочности решения различных задач в ЦУС определяет целесообразность предоставления заявкам определенных типов преимущества (приоритета) перед другими заявками (4.26). Обычно приоритеты характеризуют целыми положительными числами (1, 2, 3 ...).

Будем считать, что высшему приоритету соответствует меньшее число. Приоритеты учитываются только в момент выбора заявки из очереди, их называют относительными (ОП). Пусть каждому типу заявок соответствует свой относительный приоритет. Тогда заявка z_j будет поставлена в очередь оч_j. Если при этом процессор занят выполнением заявки z_k (низшего приоритета $k > j$), то оно будет продолжаться. Заявка z_j будет выбрана из очереди по окончании выполнения z_k, если до этого момента не поступят заявки с более высоким приоритетом.

Возможен вариант обслуживания, когда заявка, поступившая с высшим приоритетом, захватывает процессор, обслуживающий предварительную заявку. В этом случае, говорят об абсолютном приоритете (АП). Так, например, при $j < k$ заявка z_j, которая поступила в очередь ч_j, будет принята на обслуживание. При этом заявка, что обслуживалась z_k, вернется в начало очереди ч_k и будет ждать обслуживания.

Очевидно, что присвоение заявкам относительных приоритетов позволяет уменьшить τ_{wij} для заявок с высокими приоритетами за счет его увеличения для заявок с низкими приоритетами. В случае применения абсолютных приоритетов этот эффект проявляется еще сильнее.

Среднее время ожидания на *i*-м процессоре для заявок с относительным *j*-м приоритетом:

$$\tau_{wij} = \frac{\sum_{j=1}^{|\Phi|} \rho_{ij} \cdot \tau_{sij} \cdot (1 + \nu_{sij}^2)}{2 \cdot (1 - R_{i(j-1)}) \cdot (1 - R_{ij})}. \qquad (4.30)$$

Формула для определения среднего времени ожидания на *i*-м процессоре при использовании *k*-го абсолютного приоритета:

$$\tau_{wik} = \frac{\sum_{j=1}^{k} \rho_{ij} \cdot \tau_{sij} \cdot (1 + \nu_{sij}^2)}{2 \cdot (1 - R_{i(k-1)}) \cdot (1 - R_{ik})} + \frac{R_{i(k-1)} \cdot \tau_{sik}}{1 - R_{i(k-1)}} . \qquad (4.31)$$

Присвоение всем заявкам абсолютных или относительных приоритетов может привести к чрезмерному увеличению τ_{wik} для заявок с низким приоритетом. В этом случае целесообразно использование дисциплин обслуживания со смешанными приоритетами (СП).

Значения τ_{wik} для системы со смешанными приоритетами можно найти по формуле (индекс *i*, что определяет тип процессора, для R, ρ, τ, ν в формулах опущены в целях упрощения записи):

$$\tau_{wk} = \begin{cases} \dfrac{R_{k-1} \cdot \tau_{sk}}{1 - R_{k-1}} + \dfrac{\sum_{j=1}^{k} \rho_j \cdot \tau_{sj} \cdot (1 + \nu_{sj}^2)}{2 \cdot (1 - R_{k-1}) \cdot (1 - R_k)} & (k = 1,\ldots,M_1) - АП, \\ \dfrac{R_{M_1} \cdot \tau_{sk}}{1 - R_{M_1}} + \dfrac{\sum_{j=1}^{M} \rho_j \cdot \tau_{sj} \cdot (1 + \nu_{sj}^2)}{2 \cdot (1 - R_{k-1}) \cdot (1 - R_k)} & (k = M_1 + 1,\ldots,M_1 + M_2) - ВП, \\ \dfrac{R_{M_1} \cdot \tau_{sk}}{1 - R_{M_1}} + \dfrac{\sum_{j=1}^{M} \rho_j \cdot \tau_{sj} \cdot (1 + \nu_{sj}^2)}{2 \cdot (1 - R_{M_1 + M_2}) \cdot (1 - R_M)} & (k = M_1 + M_2 + 1,\ldots,M_3) - БП \end{cases} \qquad (4.32)$$

где:

M_1 – количество типов заявок из АП,

M_2 – количество типов заявок из ВП,

M_3 – количество типов заявок из БП.

Аналитические выражения (4.28), (4.29), (4.30), (4.31) и (4.32) были получены в работах [121, 18]. Система обозначений для моделей систем массового обслуживания (СМО) принята согласно работы [101].

В настоящее время нет единой методики распределения приоритетов при

дисциплине со смешанными приоритетами [121]. В простейшем случае эта задача может решаться полным перебором всех возможных вариантов распределения приоритетов и выбором наиболее приемлемого.

Для уменьшения количества переборов сначала рекомендуется переупорядочить потоки в соответствии с заданными ограничений на время ожидания, расположить их в порядке возрастания времени обслуживания. Например, если в систему поступает шесть потоков заявок 1, 2., 6, допустимое время ожидания заявок которых соответственно равен (4.26):

$\tau_{wi1}^{\text{гр}} = 1$ с, $\tau_{wi2}^{\text{гр}} = 5$ с, $\tau_{wi3}^{\text{гр}} = 2$ с, $\tau_{wi4}^{\text{гр}} = 0{,}1$ с, $\tau_{wi5}^{\text{гр}} = 0{,}5$ с, $\tau_{wi6}^{\text{гр}} = 0{,}2$ с, то в результате предварительного упорядочивания потоков получим такую последовательность 4, 6, 5, 1, 3, 2. В дальнейшем приоритеты потокам заявок должны назначаться таким образом: каждому потоку назначается приоритет не ниже, чем потокам, расположенным правее, и не выше, чем потокам, расположенным левее в указанной последовательности. В данном примере поток 4 должен быть назначен приоритет не ниже, чем всех остальных потоков; поток 6 -не ниже, чем потокам 5, 1, 3, 2, и т.д. Поток 2 должен иметь приоритет не выше, чем у всех остальных потоков.

Вид (абсолютный или относительный) и степень приоритетности каждого потока заявок по отношению к другим потоков могут выбираться любым способом. Одним из таких способов может быть целенаправленный перебор некоторых вариантов распределения приоритетов с учетом параметров потоков заявок, ограничений на время ожидания заявок и степени изменения качества обслуживания при переходе от одного варианта к другому. Целеустремленность заключается в том, чтобы каждый последующий вариант распределения приоритетов позволял удовлетворить заданным ограничениям при выбранной быстродействия процессора.

Показателем, определяющим необходимость изменения приоритета некоторого потока, может служить относительное отклонение времени ожидания τ_{wik}, полученного при данном распределении приоритетов, допустимого времени ожидания $\tau_{wik}^{\text{гр}}$:

$$\delta_{wik} = \left|\tau_{wik}^{\text{гр}} - \tau_{wik}\right| / \tau_{wik} \ (k = 1,2,...,|\Phi|), \qquad (4.33)$$

где $|\Phi|$ — число потоков заявок, поступающих в систему.

Если отклонение для одних потоков значительно отличаются от отклонений для других потоков, то необходимо изменить приоритеты этих

потоков.

Изменение приоритета потока заявок может осуществляться: 1) изменением приоритета данного потока по отношению к другим потокам; 2) изменением приоритетов других потоков по отношению к данному потоку. Распределение приоритетов удовлетворительное, если для всех потоков заявок значение δ_{wik}, примерно одинаковые.

При проектировании распределенных ИУВС задача выбора приоритетов осложняется, поскольку результат выполнения φ-транзакции в подсистеме СМО 1 может передаваться в другую подсистему СМО 5 через транзитные подсистемы СМО 3 по соответствующим каналам передачи данных СМО 2 и СМО 4. То есть необходимо рассматривать не одну простую СМО, а сеть СМО (пример такой сети показан на рис. 4.7).

Для расчета характеристик таких структур можно воспользоваться набором аналитических моделей, описанных в [101], или использовать имитационное моделирование, например, в среде JMT [171] (см. П. 2.4.4).

Несколько замечаний к рисунку 4.7.

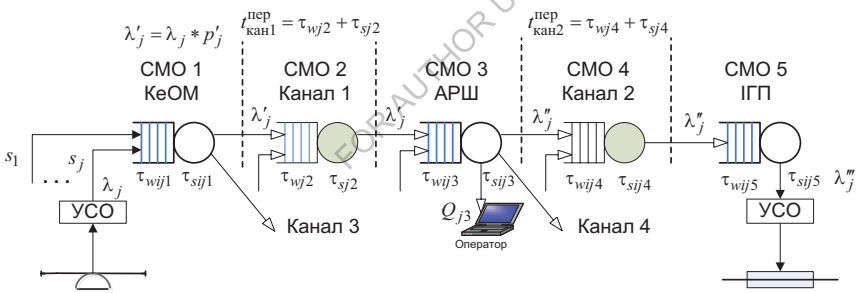


Рис. 4.7 Пример сети СМО обработки j-й φ-транзакции

В каждой подсистеме обработки данных (СМО 1, СМО 3, СМО 5) интенсивность входящего потока заявок может уменьшаться за счет разветвления транзакции.

$$\lambda'_j = \lambda_j * p'_j;\ \ \lambda''_j = \lambda_j * p''_j;\ \ \lambda'''_j = \lambda_j * p'''_j,$$

где $0 < p_j^{\text{риски}} \le 1$.

Каждый канал (1 и 2) рассматривается как СМО 2 и СМО 4, у которого время передачи определяется как $t_{\text{кан}}^{\text{пер}} = \tau_{wj(\text{кан})} + \tau_{sj(\text{кан})}$. Так как оборудование каналов на ранних стадиях еще не выбрано, то такая модель позволяет оценить требования к скорости передачи данных в канале. В данном примере только для одного *j*-го потока

$$V_{j\text{кан}1} = \frac{Q_{j3}}{t_j^{\text{пер}}} = \frac{Q_{j3}}{t_{j3}^{\text{гр}} - \tau_{wij1} - \tau_{sij1}}. \qquad (4.34)$$

Очевидно, что сумма всех потоков позволяет оценить необходимую скорость передачи данных для всех видов информационных потоков (сообщений и управляющих действий) всех транзакций, использующих соответствующий канал *k*.

$$V_k = \sum_{j \in K_k} V_j . \qquad (4.35)$$

Многократное решение задачи выбора приоритетов в процессе проектирования можно облегчить, используя разработанную программу PRIORY.

5. Усовершенствование методов анализа и расчета надежности систем с нечеткими параметрами

5.1 Метод анализа надежности нечетких систем с использованием теории размытых множеств

Выполненные исследования (глава 3) свидетельствуют о важности задачи моделирования надежности при проектировании систем [75]. Обычно надежное поведение системы полностью описывается вероятностными характеристиками. Тем не менее, из-за неточности и неопределенности исходных данных, оценка точных значений вероятностей во многих системах представляется сложной задачей [178]. В последние годы некоторые исследователи сосредоточены на использовании теории нечетких множеств [214] для анализа надежности нечетких системы [178, 183, 179. В [174] представлены следующие два предположения:

(1) Fuzzy-состояние: в любое время система может находиться либо в состоянии нечеткой трудоспособности или в состоянии нечеткой отказа.

(2) Предположение о возможности: поведение системы может быть полностью охарактеризованы по мере возможности.

В работе [178] представлен анализ работ, связанных с введением в инженерию систем с отказами и использованием в ней нечеткой методологии, анализом надежности нечеткой системы с использованием нечетких числовых арифметических операций, анализом надежности нечеткой системы, основанный на нечетких временных рядах и α- отсечении в операциях над нечеткими числами, нечетким анализом надежности системы с использованием интервала доверия и других известных подходов и методов в данной области.

В этом подпункте представляется новый табличный метод анализа надежности нечеткой системы с использованием размытых множеств [183], [179], где надежностный параметры компонентов системы представляются размытыми множествами, которые определяются в универсуме [0, 1]. Степень принадлежности элемента x в размытом множестве представляется размытым значением $[t_x, 1-f_x]$ в [0, 1], где t_x указывает на степень истины, f_x указывает на степень ошибочности, $1-t_x-f_x$ указывает на неизвестную часть $0 \le t_x \le 1-f_x \le 1$, и $t_x+f_x \le 1$. Понятие размытых множеств похоже на интуитивные нечеткие множества [169], и оба они являются обобщением

понятия нечетких множеств [214]. Предлагаемый метод табличного моделирования и анализа надежности нечеткой системы обладает наглядностью, гибкостью и простой интеллектуальной формой.

Базовые концепты размытых множеств. В 1965 году Заде предложил теорию нечетких множеств [214] или нечеткое множество классов с нечеткими границами. Пусть имеем универсум U, состоящий из n значений $U=\{u_1,u_2, ..., u_n\}$. Степень принадлежности элемента u_i нечеткому множеству представлены реальными значениями между нулем и единицей, где $u_i \in U$. В [183] Гау отметил, что это одно значение степени принадлежности сочетает в себе доказательство, что $u_i \in U$, и демонстрирует противоречия, что $u_i \in U$. Это не является доказательством, что $u_i \in U$, и доказательством против $u_i \in U$ соответственно, и оно не указывает, сколько там каждого из них. Кроме того, Гау также отметил, что одно значение степени ничего не говорит о его точности и представил в [183] основные понятия размытых множества. В [179] предложенные правила выполнения арифметических операций над размытыми множествами. пусть U универсум $U=\{u_1,u_2, ..., u_n\}$, который описывается элементами. Размытое множество $\tilde{A}$ на универсуме U характеризуется функцией принадлежности истина (ФПИ) $t_{\tilde{A}}, t_{\tilde{A}} : U \to [0,1]$, и функцией принадлежности ошибочности (ФПО) $f_{\tilde{A}}$, $f_{\tilde{A}} : U \to [0,1]$, где $t_{\tilde{A}}(u_i)$ нижняя граница функции принадлежности u_i, что является доказательством для значения u_i, $f_{\tilde{A}}(u_i)$ - нижняя граница возражения u_i, что является доказательством против значения u_i, и $t_{\tilde{A}}(u_i) + f_{\tilde{A}}(u_i) \le 1$. Значение функции принадлежности u_i размытого множества $\tilde{A}$ является ограниченный подинтервал $[t_{\tilde{A}}(u_i), 1 - f_{\tilde{A}}(u_i)]$ в [0, 1]. Размытая величина $[t_{\tilde{A}}(u_i), 1 - f_{\tilde{A}}(u_i)]$ показывает, что точное значение функции принадлежности $\mu_{\tilde{A}}(u_i)$ для u_i ограничено диапазоном $t_{\tilde{A}}(u_i) \le \mu_{\tilde{A}}(u_i) \le 1 - f_{\tilde{A}}(u_i)$, где $t_{\tilde{A}}(u_i) + f_{\tilde{A}}(u_i) \le 1$. Пример размытого множества $\tilde{A}$ на универсуме U показано на рис. 5.1.

Если универс U является дискретным множеством, то размытое множество $\tilde{A}$ на универсе U может быть представлено как

$$\tilde{A} = \sum_{i=1}^{n} [t_{\tilde{A}}(u_i), 1 - f_{\tilde{A}}(u_i)] / u_i, \; u_i \in U. \qquad (5.23)$$

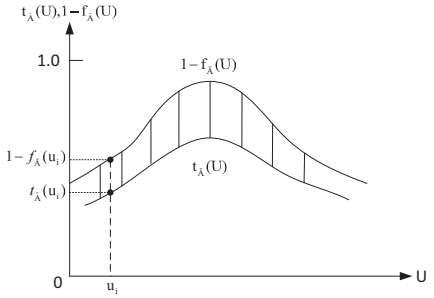


Рис. 5.1 Размытое множество

Если универс U является непрерывным множеством, то размытое множество $\tilde{A}$ на универсе U может быть представлено как

$$\tilde{A}=\int_U [t_{\tilde{A}}(u_i), 1-f_{\tilde{A}}(u_i)] / u_i, u_i \in U \qquad (5.24)$$

Определение 1. Пусть $\tilde{A}$ размытое множество на универсе U из ФПИ $t_{\tilde{A}}$ и ФПО $f_{\tilde{A}}$ соответственно. Размытое множество $\tilde{A}$ является выпуклой тогда и только тогда, когда для всех u_1, u_2 на U,

$$t_{\tilde{A}}(\lambda u_1 + (1-\lambda)u_2) \geq \min(t_{\tilde{A}}(u_1), t_{\tilde{A}}(u_2)), \qquad (5.25)$$

$$1- f_{\tilde{A}}(\lambda u_1 + (1-\lambda)u_2) \geq \min(1- f_{\tilde{A}}(u_1), 1- f_{\tilde{A}}(u_2)), \text{ где } \lambda \in [0,1]. \qquad (5.26)$$

Определение 2. Размытое множество $\tilde{A}$ на универсе U называется нормальным размытым множеством, если $\exists\, u_i \in U$ для которого $1-f_{\tilde{A}}(u_i)=1$. В этом случае $f_{\tilde{A}}(u_i)=0$.

Определение 3. Размытым числом является такое размытое подмножество на универсале U, которое является как выпуклой, так и нормальной.

Далее, введем основные арифметические операции над треугольными размытыми множествами [179].

Рассмотрим треугольное размытое множество $\tilde{A}$, показаное на рис. 5.2, где треугольное размытое множество $\tilde{A}$ может быть параметризованно кортежем $<[(a,b,c);\mu_1], [(a,b,c);\mu_2]>$. Для удобства кортеж $<[(a,b,c);\mu_1], [(a,b,c);\mu_2]>$ будем представлять кортежем $<[(a,b,c);\mu_1,\mu_2]>$, где $0 \leq \mu_1 \leq \mu_2 \leq 1$.

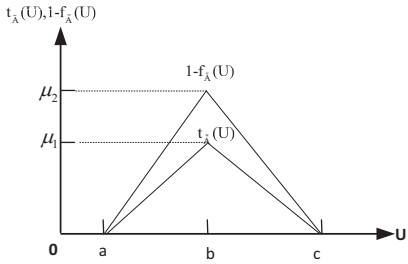


Рис. 5.2 Треугольное размытое множество

Теперь перейдем к арифметических операций между треугольными размытыми множествами.

Случай 1. Рассмотрим треугольные размытые множества $\tilde{A}$ и $\tilde{B}$, показанные на рис. 5.3, где

$\tilde{A}=<[(a_1,b_1,c_1);\mu_1],[(a_1,b_1,c_1);\mu_2]> = <[(a_1,b_1,c_1);\mu_1;\mu_2]>,$

$\tilde{B}=<[(a_2,b_2,c_2);\mu_1],[(a_2,b_2,c_2);\mu_2]> = <[(a_2,b_2,c_2);\mu_1;\mu_2]>,$

і $0 \le \mu_1 \le \mu_2 \le 1$.

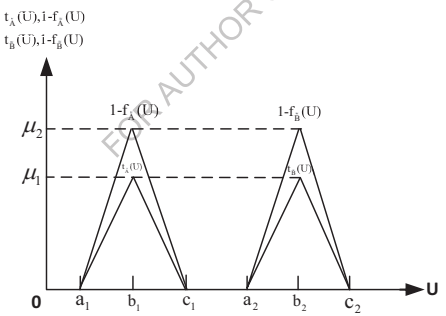


Рис. 5.3 Размытые треугольные множества $\tilde{A}$ и $\tilde{B}$ (случай 1)

Арифметические операции между треугольными размытыми множествами $\tilde{A}$ и $\tilde{B}$ определяются следующим образом.

Сложение ($\oplus$):

$$\begin{aligned}\tilde{A} \oplus \tilde{B} &= <[(a_1,b_1,c_1);\mu_1],\ [(a_1,b_1,c_1);\mu_2]> \oplus \\ &\oplus <[(a_2,b_2,c_2);\mu_1],\ [(a_2,b_2,c_2);\mu_2]> = \\ &= <[(a_1+a_2,b_1+b_2,c_1+c_2);\mu_1],\ [(a_1+a_2,b_1+b_2,c_1+c_2);\mu_2]> \\ &= <[(a_1+a_2,b_1+b_2,c_1+c_2);\mu_1;\mu_2]>.\end{aligned} \quad (5.27)$$

Вычитание ($\ominus$):

$$\begin{aligned}\tilde{B}\ominus\tilde{A}=&<[(a_2,b_2,c_2);\mu_1],[(a_2,b_2,c_2);\mu_2]>\ominus\\ &\ominus<[(a_1,b_1,c_1);\mu_1],[(a_1,b_1,c_1);\mu_2]>=\\ &=<[(a_2-c_1,b_2-b_1,c_2-a_1);\mu_1],[(a_2-c_1,b_2-b_1,c_2-a_1);\mu_2]>=\\ &=<[(a_2-c_1,b_2-b_1,c_2-a_1);\mu_1;\mu_2]>.\end{aligned} \quad (5.28)$$

Умножение ($\otimes$):

$$\begin{aligned}\tilde{A}\otimes\tilde{B}=&<[(a_1,b_1,c_1);\mu_1],[(a_1,b_1,c_1);\mu_2]>\otimes\\ &\otimes<[(a_2,b_2,c_2);\mu_1],[(a_2,b_2,c_2);\mu_2]>=\\ &=<[(a_1\times a_2,b_1\times b_2,c_1\times c_2);\mu_1],[(a_1\times a_2,b_1\times b_2,c_1\times c_2);\mu_2]>=\\ &=<[(a_1\times a_2,b_1\times b_2,c_1\times c_2);\mu_1;\mu_2]>.\end{aligned} \quad (5.29)$$

Деление ($\oslash$):

$$\begin{aligned}\tilde{B}\oslash\tilde{A}=&<[(a_2,b_2,c_2);\mu_1],[(a_2,b_2,c_2);\mu_2]>\oslash\\ &\oslash<[(a_1,b_1,c_1);\mu_1],[(a_1,b_1,c_1);\mu_2]>=\\ &=<[(a_2/c_1,b_2/b_1,c_2/a_1);\mu_1],[(a_2/c_1,b_2/b_1,c_2/a_1);\mu_2]>=\\ &=<[(a_2/c_1,b_2/b_1,c_2/a_1);\mu_1;\mu_2]>.\end{aligned} \quad (5.30)$$

Случай 2. Рассмотрим треугольные размытые множества $\tilde{A}$ и $\tilde{B}$, показанные на рис. 5.4, где

$$\tilde{A}=<[(a_1,b_1,c_1);\mu_1],[(a_1,b_1,c_1);\mu_2]>,$$

$$\tilde{B}=<[(a_2,b_2,c_2);\mu_3],[(a_2,b_2,c_2);\mu_4]>,$$

$$\text{i } 0\le\mu_3\le\mu_1\le\mu_4\le\mu_2\le 1.$$

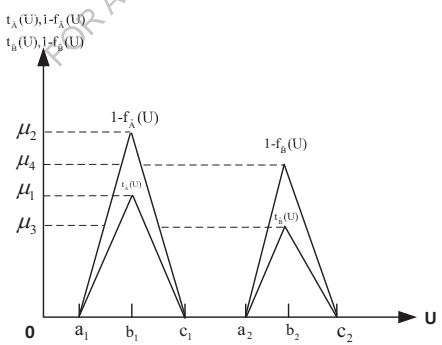


Рис. 5.4 Размытые треугольные множества $\tilde{A}$ и $\tilde{B}$ (випадок 2)

Арифметические операции в этом случае между треугольными размытыми множествами $\tilde{A}$ и $\tilde{B}$ определяются следующим образом.

Сложение:

$$\begin{aligned}\tilde{A}\oplus\tilde{B}=&<[(a_1,b_1,c_1);\mu_1],[(a_1,b_1,c_1);\mu_2]>\oplus\otimes\\ &\oplus<[(a_2,b_2,c_2);\mu_3],[(a_2,b_2,c_2);\mu_4]>=\\ &=<[(a_1+a_2,b_1+b_2,c_1+c_2);\min(\mu_1,\mu_3)],\\ &\quad[(a_1+a_2,b_1+b_2,c_1+c_2);\min(\mu_2,\mu_4)]>.\end{aligned}\tag{5.31}$$

Вычитание:

$$\begin{aligned}\tilde{B}\ominus\tilde{A}=&<[(a_2,b_2,c_2);\mu_3],[(a_2,b_2,c_2);\mu_4]>\ominus\\ &\ominus<[(a_1,b_1,c_1);\mu_1],[(a_1,b_1,c_1);\mu_2]>=\\ &=<[(a_2-c_1,b_2-b_1,c_2-a_1);\min(\mu_1-\mu_3)],\\ &\quad[(a_2-c_1,b_2-b_1,c_2-a_1);\min(\mu_2-\mu_4)]>.\end{aligned}\tag{5.32}$$

Умножение:

$$\begin{aligned}\tilde{A}\otimes\tilde{B}=&<[(a_1,b_1,c_1);\mu_1],[(a_1,b_1,c_1);\mu_2]>\otimes\\ &\otimes<[(a_2,b_2,c_2);\mu_3],[(a_2,b_2,c_2);\mu_4]>=\\ &=<[(a_1\times a_2,b_1\times b_2,c_1\times c_2);\min(\mu_1,\mu_3)],\\ &\quad[(a_1\times a_2,b_1\times b_2,c_1\times c_2);\min(\mu_2,\mu_4)]>.\end{aligned}\tag{5.33}$$

Деление:

$$\begin{aligned}\tilde{B}\oslash\tilde{A}=&<[(a_2,b_2,c_2);\mu_3],[(a_2,b_2,c_2);\mu_4]>\oslash\\ &\oslash<[(a_1,b_1,c_1);\mu_1],[(a_1,b_1,c_1);\mu_2]>=\\ &=<[(a_2/c_1,b_2/b_1,c_2/a_1);\min(\mu_1,\mu_3)],\\ &\quad[(a_2/c_1,b_2/b_1,c_2/a_1);\min(\mu_2,\mu_4)]>.\end{aligned}\tag{5.34}$$

Анализ надежности нечетких систем на основе размытых множеств. Здесь рассмотрим новый метод анализа надежности нечетких систем на основе теории размытых множеств, где надежность компонентов системы представляются размытыми множествами, определенными на универсале [0, 1].

Последовательные структуры. Начнем с последовательно надежностной схемы соединений компонентов системы, как показано на рис. 5.10, где надежность $\tilde{R}_i$ компоненты P_i представляется размытым множеством $<[(a_i,b_i,c_i);\mu_{i1};\mu_{i2}]>$, где $0\le\mu_{i1}\le\mu_{i2}\le 1$, а $1\le i\le n$. Тогда надежность $\tilde{R}$ последовательной системы, показанной на рис. 5.5, может быть рассчитана по формуле

$$\begin{aligned}&\tilde{R}=\tilde{R}_1\otimes\tilde{R}_2\otimes\ldots\otimes\tilde{R}_n=\\ &=<[(a_1,b_1,c_1);\mu_{11};\mu_{12}]>\otimes<[(a_2,b_2,c_2);\mu_{21};\mu_{22}]>\otimes\ldots\\ &\ldots\otimes<[(a_n,b_n,c_n);\mu_{n1};\mu_{n2}]>=\\ &=<[(\overset{n}{\underset{i=1}{\otimes}}a_i,\overset{n}{\underset{i=1}{\otimes}}b_i,\overset{n}{\underset{i=1}{\otimes}}c_i);\min(\mu_{11},\mu_{21},\ldots,\mu_{n1});\min(\mu_{12},\mu_{22},\ldots,\mu_{n2})]>.\end{aligned}\tag{5.35}$$

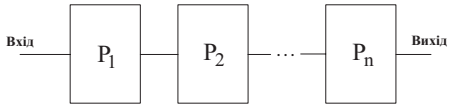


Рис. 5.5 Структура последовательной системы

Пример 1.

Рассмотрим пример цепи с 3-х последовательно соединенных элементов. Исходные значения параметров и результат приведены на рис. 5.6.

	A	B	C	D	E	F	G	H	I	J	K	L	M
1													
2								ПРОИЗВЕД(C4:E4)					
3			R1	R2	R2	R	m_R						
4		a1	0,85	0,87	0,7	0,51765	0						
5		b1	0,9	0,95	0,92	0,7866	0,5						
6		c1	0,95	0,97	0,98	0,90307	0						
7		a2	0,85	0,87	0,7	0,51765	0						
8		b2	0,9	0,95	0,92	0,7866	0,3						
9		c2	0,95	0,97	0,98	0,90307	0						
10		m_1_max	0,5	0,6	0,7	0,5							
11		m_2_min	0,4	0,3	0,25	0,25							
12								МИН(C10:E10)					

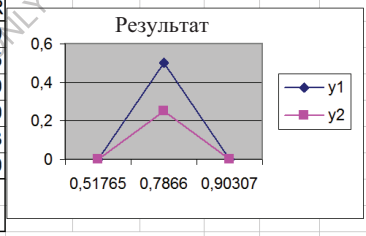


Рис. 5.6 Характеристики последовательной структуры

Параллельные структуры (системы с резервированием). Теперь рассмотрим параллельную систему, изображенную на рис. 5.7, где надежность $\tilde{R}_i$ компоненты P_i представляется размытым множеством $<[(a_i,b_i,c_i);\mu_{i1};\mu_{i2}]>$, де $0 \le \mu_{i1} \le \mu_{i2} \le 1$, а $1 \le i \le n$. Тогда надежность $\tilde{R}$ параллельной системы, показанной на рис. 5.11, может быть рассчитана по формуле

$$\tilde{R}=1 \ominus \overset{n}{\underset{i=1}{\otimes}} (1 \ominus \tilde{R}_i) = \qquad (5.36)$$

$$= 1 \ominus (1 \ominus < [(a_1, b_1, c_1); \mu_{11}; \mu_{12}] >) \otimes (1 \ominus < [(a_2, b_2, c_2); \mu_{21}; \mu_{22}] >) \otimes \ldots$$
$$\ldots \otimes (1 \ominus < [(a_n, b_n, c_n); \mu_{n1}; \mu_{n2}] >) =$$
$$= 1 \ominus < [(1 - c_1, 1 - b_1, 1 - a_1); \mu_{11}; \mu_{12}] > \otimes < [(1 - c_2, 1 - b_2, 1 - a_2); \mu_{21}; \mu_{22}] > \otimes \ldots$$
$$\ldots \otimes < [(1 - c_n, 1 - b_n, 1 - a_n); \mu_{n1}; \mu_{n2}] > =$$
$$= 1 \ominus < (\overset{n}{\underset{i=1}{\otimes}}(1 - c_i), \overset{n}{\underset{i=1}{\otimes}}(1 - b_i), \overset{n}{\underset{i=1}{\otimes}}(1 - a_i));$$
$$\min(\mu_{11}; \mu_{21}; \ldots, \mu_{n1}); \min(\mu_{12}; \mu_{22}; \ldots, \mu_{n2})] >=$$
$$= < [(1 - \overset{n}{\underset{i=1}{\otimes}}(1 - a_i), 1 - \overset{n}{\underset{i=1}{\otimes}}(1 - b_i), 1 - \overset{n}{\underset{i=1}{\otimes}}(1 - c_i));$$
$$\min(\mu_{11}; \mu_{21}; \ldots, \mu_{n1}); \min(\mu_{12}; \mu_{22}; \ldots, \mu_{n2})] >.$$

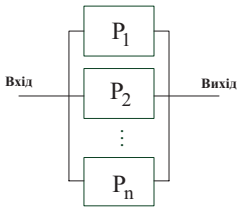


Рис. 5.7 Структура параллельной системы

Пример 2.

Рассмотрим пример цепи с 3-х параллельно соединенных элементов. Исходные значения параметров и результаты приведены в таблице на рис. 5.8.

	R1	R2	R3	1-C3			F3*G3*H3	1-I3	m_R
a1	0,85	0,87	0,7	0,15	0,13	0,3	0,00585	0,99415	0
b1	0,9	0,95	0,92	0,1	0,05	0,08	0,0004	0,9996	0,5
c1	0,95	0,97	0,98	0,05	0,03	0,02	0,000030	0,99997	0
a2	0,85	0,87	0,7	0,15	0,13	0,3	0,00585	0,99415	0
b2	0,9	0,95	0,92	0,1	0,05	0,08	0,0004	0,9996	0,25
c2	0,95	0,97	0,98	0,05	0,03	0,02	0,000030	0,99997	0
m_1_m	0,5	0,6	0,7				0,5		
m_2_m	0,4	0,3	0,25				0,25		

МИН(C10:E10)

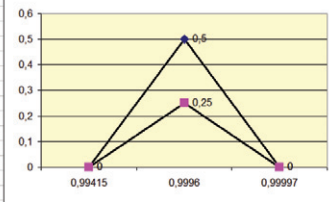


Рис. 5.8 Характеристики системы с тройным резервированием

Графическое представление исходных данных и результатов надежностный моделирования приведены на рисунке 5.9.

Комбинированные структуры. Рассмотрим пример, иллюстрирующий применение описанного метода для анализа надежности последовательно-параллельной структуры нечеткой системы.

Пример 3. Исследуем систему, показанную на рис. 5.10, где надежность компонентов P_1, P_2, P_3 и P_4 являются соответственно $\tilde{R}_1, \tilde{R}_2, \tilde{R}_3$ і $\tilde{R}_4$, где

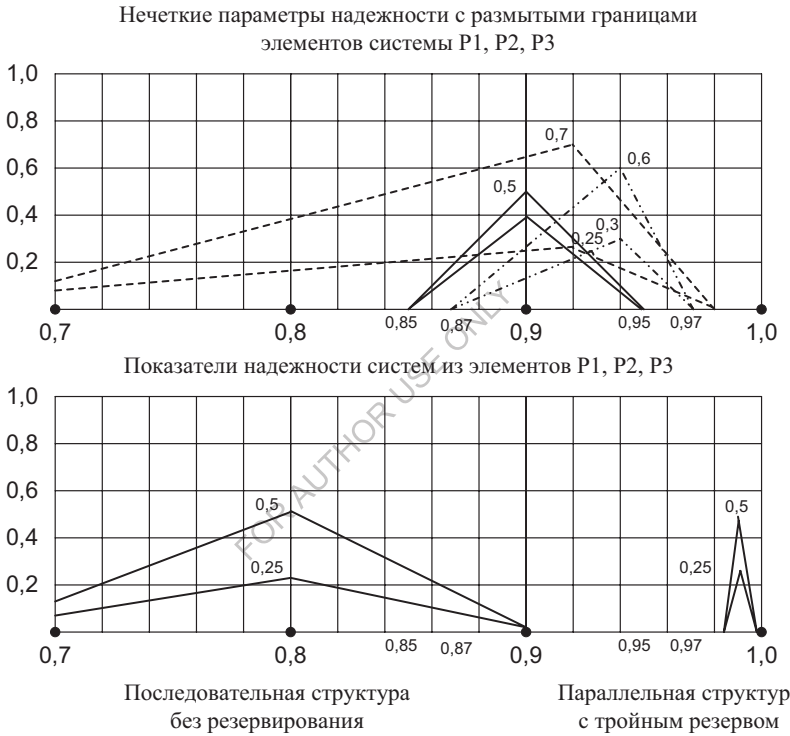


Рис. 5.9 Графические результаты решения задач (рис. 5.7 і рис. 5.9)

$\tilde{R}_1 = <[(a_1,b_1,c_1);\mu_{11};\mu_{12}]>,$
$\tilde{R}_2 = <[(a_2,b_2,c_2);\mu_{21};\mu_{22}]>,$
$\tilde{R}_3 = <[(a_3,b_3,c_3);\mu_{31};\mu_{32}]>,$
$\tilde{R}_4 = <[(a_4,b_4,c_4);\mu_{41};\mu_{42}]>,$

$0 \le \mu_{i1} \le \mu_{i2} \le 1$ та $1 \le i \le 4$.

Основываясь на предыдущих определениях, мы можем оценить надежность $\tilde{R}$ системы, приведенной на рис. 5.10, с помощью следующих выражений:

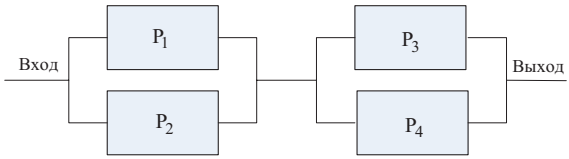


Рис. 5.10 Последовательно-параллельная структура

$$\tilde{R} = [1 \ominus (1 \ominus \tilde{R}_1) \otimes (1 \ominus \tilde{R}_2)] \otimes [1 \ominus (1 \ominus \tilde{R}_3) \otimes (1 \ominus \tilde{R}_4)] =$$

$$= [1 \ominus (1 \ominus < [(a_1, b_1, c_1); \mu_{11}; \mu_{12}] >) \otimes (1 \ominus < [(a_2, b_2, c_2); \mu_{21}; \mu_{22}] >) \otimes$$
$$\otimes [1 \ominus (1 \ominus < [(a_3, b_3, c_3); \mu_{31}; \mu_{32}] >) \otimes (1 \ominus < [(a_4, b_4, c_4); \mu_{41}; \mu_{42}] >)] =$$

$$= [1 \ominus < [(1 - c_1, 1 - b_1, 1 - a_1); \mu_{11};\ \mu_{12}] > \otimes < [(1 - c_2, 1 - b_2, 1 - a_2); \mu_{21}; \mu_{22}] >] \otimes$$
$$\otimes [1 \ominus < [(1 - c_3, 1 - b_3, 1 - a_3); \mu_{31}; \mu_{32}] > \otimes < [(1 - c_4, 1 - b_4, 1 - a_4); \mu_{41}; \mu_{42}] >] =$$

$$= [1 \ominus < [((1 - c_1)(1 - c_2), (1 - b_1)(1 - b_2), (1 - a_1)(1 - a_2));$$
$$\min(\mu_{11}; \mu_{21});\ \min(\mu_{12}, \mu_{22})] >] \otimes$$
$$\otimes [1 \ominus < [((1 - c_3)(1 - c_4), (1 - b_3)(1 - b_4), (1 - a_3)(1 - a_4));$$
$$\min(\mu_{31}; \mu_{41});\ \min(\mu_{32}, \mu_{42})] >] =$$

$$= < [1 - (1 - a_1)(1 - a_2), 1 - (1 - b_1)(1 - b_2),\ 1 - (1 - c_1)(1 - c_2));$$
$$\min(\mu_{11}; \mu_{21}); \min(\mu_{12}, \mu_{22})] >] \otimes$$
$$\otimes < [(1 - (1 - a_3)(1 - a_4), 1 - (1 - b_3)(1 - b_4), 1 - (1 - c_3)(1 - c_4));$$
$$\min(\mu_{31}; \mu_{41});\ \min(\mu_{32}, \mu_{42})] > =$$

$$= < [(a_1 + a_2 - a_1 a_2,\ b_1 + b_2 - b_1 b_2,\ c_1 + c_2 - c_1 c_2);$$
$$\min(\mu_{11}, \mu_{21});\ \min(\mu_{12}, \mu_{22})] > \otimes$$
$$\otimes < [(a_3 + a_4 - a_3 a_4,\ b_3 + b_4 - b_3 b_4,\ c_3 + c_4 - c_3 c_4);$$
$$\min(\mu_{31}, \mu_{41});\ \min(\mu_{32}, \mu_{42})] > =$$

$$= < [((a_1 + a_2 - a_1 a_2)(a_3 + a_4 - a_3 a_4), (b_1 + b_2 - b_1 b_2)(b_3 + b_4 - b_3 b_4),$$
$$(c_1 + c_2 - c_1 c_2)(c_3 + c_4 - c_3 c_4));$$
$$\min(\mu_{11}, \mu_{21}, \mu_{31}, \mu_{41});\ \min(\mu_{12}, \mu_{22}, \mu_{32}, \mu_{42})] > =$$

$$= < [(a_1 a_3 + a_1 a_4 - a_1 a_3 a_4 + a_2 a_3 + a_2 a_4 - a_2 a_3 a_4 - a_1 a_2 a_3 - a_1 a_2 a_4 + a_1 a_2 a_3 a_4,$$
$$b_1 b_3 + b_1 b_4 - b_1 b_3 b_4 + b_2 b_3 + b_2 b_4 - b_2 b_3 b_4 - b_1 b_2 b_3 - b_1 b_2 b_4 + b_1 b_2 b_3 b_4,$$
$$c_1 c_3 + c_1 c_4 - c_1 c_3 c_4 + c_2 c_3 + c_2 c_4 - c_2 c_3 c_4 - c_1 c_2 c_3 - c_1 c_2 c_4 + c_1 c_2 c_3 c_4);$$
$$\min(\mu_{11}, \mu_{21}, \mu_{31}, \mu_{41});\ \min(\mu_{12}, \mu_{22}, \mu_{32}, \mu_{42})] >.$$

Математические выражения для оценки надежности нечетких систем с размытыми параметрами получить достаточно сложно, особенно при построении комбинированных структур. Более эффективным и гибким решением является использование табличного описания взаимосвязанных подсистем. Например, структуру, изображенную на рис. 5.10, можно представить в виде набора таблиц, приведенных на рис. 5.7 и 5.9. В результате получаем эффективную, персонализированную модель для надежностного моделирования нечетких систем с размытыми параметрами (рис. 5.11).

Предложенный метод табличного описания нечетких систем, элементы

которых описываются треугольными функциями принадлежности с размытыми границами, позволяет моделировать характеристики надежности подобных систем путем построения взаимосвязанных таблиц для последовательных и параллельных структур без выполнения сложных математических преобразований, снижает вероятность ошибок, сокращает время построения моделей и моделирование систем в целом.

U	R1	R2	Промежуточные результаты			R12	m_R
a1	0,75	0,75	0,25	0,25	0,0625	0,9375	0
b1	0,8	0,99	0,2	0,01	0,002	0,998	0,7
c1	0,92	0,998	0,08	0,002	0,00016	0,9998	0
a2	0,75	0,75	0,25	0,25	0,0625	0,9375	0
b2	0,8	0,99	0,2	0,01	0,002	0,998	0,2
c2	0,92	0,998	0,08	0,002	0,00016	0,9998	0
m_1_n	0,7	0,8			0,7		
m_2_n	0,25	0,2			0,2		

U	R3	R4	Промежуточные результаты			R34	m_R
a1	0,8	0,87	0,2	0,13	0,026	0,974	0
b1	0,85	0,99	0,15	0,01	0,0015	0,9985	0,8
c1	0,9	0,995	0,1	0,005	0,0005	0,9995	0
a2	0,8	0,87	0,2	0,13	0,026	0,974	0
b2	0,85	0,99	0,15	0,01	0,0015	0,9985	0,1
c2	0,9	0,995	0,1	0,005	0,0005	0,9995	0
m_1_n	0,8	0,9			0,8		
m_2_n	0,18	0,1			0,1		

U	R12	R34	R	m_R
a1	0,9375	0,974	0,9131	0
b1	0,998	0,9985	0,9965	0,7
c1	0,9998	0,9995	0,9993	0
a2	0,9375	0,974	0,9131	0
b2	0,998	0,9985	0,9965	0,1
c2	0,9998	0,9995	0,9993	0
m_1_n	0,7	0,8	0,7	
m_2_n	0,2	0,1	0,1	

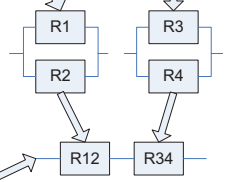


Рис. 5.11 Табличное моделирование нечетких систем

5.2 Метод использования различных видов размытых множеств для оценки надежности нечетких систем

До сих пор в литературе обсуждаются вопросы выполнения арифметических операций между различными типами размытых множеств [75, 195, 214, 207, 174, 183, 156, 178]. Однако, как и ранее при анализе надежности нечетких систем предполагается, что надежность всех компонентов системы имеет один и тот же вид функций принадлежности [75]. Однако, в практических задачах такая ситуация встречается редко. Итак, необходим

метод, который позволял бы находить надежность нечетких систем с компонентами, которые имеют различные типы функций принадлежности.

В методе предлагается унифицированное описание показателей надежности нечетких систем с размытыми границами и различными видами функций принадлежности в виде кортежа $<[a_1,b_{11},b_{12},c_1];[a_2,b_{21},b_{22},c_2];\mu_1,\mu_2>$.

Для предложенной структуры кортежа рассматривается новый алгоритм выполнения различных арифметических операций между различными видами расплывчатых множеств и предлагаются табличные методы анализа надежности различных нечетких систем. Исследование обобщают известные работы в этой области [75, 195, 156, 178, 183, 175].

В процессе проектирования и технического перевооружения сортировочных станций, при использовании новых устройств, мы не можем в численном виде выразить их технические характеристики и влияние различных факторов на процесс функционирования сортировочной системы. Эти факторы обычно имеют некоторую неопределенность и языковые неточности в их определении, что связано со следующими обстоятельствами.

Большинство систем управления на сортировочных станциях и горках являются слишком дорогими и сложными, и экспериментально измерить их характеристики в реальных условиях эксплуатации достаточно сложно, так как это связано с производственными потерями. Поэтому пользуются экспертным оценкам для выявления сбоев в системах. Однако, оценки экспертов обычно являются неопределенными и ситуации описываются различными синтаксическими конструкциями.

Нормальное или ненормальное состояние системы нельзя точно определить, так как в ней реализуются функции при некоторых ограничениях и не гарантируется 100% надежность системы и ее устройств.

Станционные системы автоматизации включают устройства, построенные на разной элементной базе - механические, релейные, электронные. Мы не можем исключить любую возможность сбоев системы, включая системы электроснабжения, природные условия и, конечно, человеческий фактор.

Поэтому предлагается использовать для анализа надежности нечетких систем автоматизации сортировочных станций размытые множества, которые помогут решить различные проблемы. В данном случае эксперты должны только указать диапазоны уровней доверия, соответствующие состояниям отказа в устройствах системы.

Определение размытых множеств изложенные в предыдущем пункте [183].

Рассмотрим алгоритм выполнения точных арифметических операций между различными типами размытых множеств $\tilde{A}_1, \tilde{A}_2, ..., \tilde{A}_n$ [195].

Для унификации описания различных треугольных и трапециевидных функций введем следующий кортеж:

$$< [a_1, b_{11}, b_{12}, c_1]; [a_2, b_{21}, b_{22}, c_2]; \mu_1, \mu_2 >, \qquad (5.37)$$

где $a_1, b_{11}, b_{12}, c_1, a_2, b_{21}, b_{22}, c_2 \in U$ и $\mu_1, \mu_2 \in [0,1]$.

Он соответствует наиболее сложному виду трапециевидных функций с несовпадающими показателями надежности в конечных и экстремальных точках. В случае треугольных функций $b_{11} = b_{12}$ и $b_{21} = b_{22}$. Возможные варианты описания функций этим кортежем представлены в примерах и сведены в таблицу 5.1.

Шаг 1

Найти значение

$$\alpha_i = \sup_{i=0,1,...,n} (t_{\tilde{A}_i}(U)) = \max_{u_i \in U} \{t_{\tilde{A}}(u_i)\},$$

$$\beta_i = \sup_{i=0,1,...,n} (1 - f_{\tilde{A}_i}(U)) = \max_{u_i \in U} \{1 - f_{\tilde{A}}(u_i)\}.$$

Шаг 2

Найти $\forall i = 0,1,2,...,n$ значение $\alpha = \min(\alpha_i)$ и $\beta = \min(\beta_i)$.

Шаг 3

Найти интервалы определенных значений $t_{\tilde{A}_i}(U)$, принадлежащих диапазону $[0, \alpha]$. Пусть интервал для $t_{\tilde{A}_i}(U) = p$, $0 \le p \le \alpha$ є $[u_{i1}, u_{i2}]$, где $u_{i1}, u_{i2} \in U$.

Шаг 4

Найти интервалы определенных значений $1 - f_{\tilde{A}_i}(U)$, принадлежащих диапазону $[0, \beta]$. Пусть интервал для $1 - f_{\tilde{A}_i}(U) = q$, $0 \le q \le \beta$ є $[u'_{i1}, u'_{i2}]$, где $u'_{i1}, u'_{i2} \in U$.

Шаг 5

Определим сложение, умножение и вычитание размытых множеств для

$t_{\tilde{A}}(U) = p$ как $[\sum_{i=1}^{n} u_{i1}, \sum_{i=1}^{n} u_{i2}]$, $[\prod_{i=1}^{n} u_{i1}, \prod_{i=1}^{n} u_{i2}]$ и $[u_{11}-(\sum_{i=2}^{n} u_{i2}), u_{12}-(\sum_{i=2}^{n} u_{i1})]$

соответственно, где $\tilde{A}$ представляет результат - размытое множество.

Шаг 6

Определим сложение, умножение и вычитание размытых множеств для

$1 - f_{\tilde{A}}(U) = q$ как $[\sum_{i=1}^{n} u'_{i1}, \sum_{i=1}^{n} u'_{i2}]$, $[\prod_{i=1}^{n} u'_{i1}, \prod_{i=1}^{n} u'_{i2}]$ и

$[u'_{11}-(\sum_{i=2}^{n} u'_{i2}), u'_{12}-(\sum_{i=2}^{n} u'_{i1})]$ соответственно.

Шаг 7

Нарисовать функции принадлежности результирующих размытых множеств после нахождения интервалов для определенных значений p (включая 0 и α) и q (включая 0 и β).

Введеная избыточность позволяет формализовать процесс вычисления результирующих показателей надежности и построение соответствующих функций принадлежности по заданным значениям $t_{\tilde{A}}(u_i)$ и $1 - f_{\tilde{A}}(u_i)$, то есть μ_1 и μ_2.

Для левой (растущей) части функции

$$u_i = a_1 + \frac{t_{\tilde{A}}(u_i)(b_{11} - a_1)}{\alpha_i}, \tag{5.38}$$

$$u_i = a_1 + \frac{(1 - f_{\tilde{A}}(u_i))(b_{11} - a_1)}{\beta_i}, \tag{5.39}$$

Для правой (ниспадающей) части функции

$$u_i = c_2 - \frac{t_{\tilde{A}}(u_i)(c_2 - b_{22})}{\alpha_i}, \tag{5.40}$$

$$u_i = c_2 - \frac{(1 - f_{\tilde{A}}(u_i))(c_2 - b_{22})}{\beta_i}, \tag{5.41}$$

Рассмотрим применение разработанного описания размытых нечетких множеств и предложенного алгоритма для анализа надежности последовательно, параллельно, параллельно-последовательных и последовательно-параллельных нечетких систем, где надежностные параметры

компонентов системы представлены различными типами размытых множеств, определяемых на универсале [0, 1].

Последовательная система. Рассмотрим последовательную систему, которая состоит из "n" компонентов и показана на рис. 5.17.

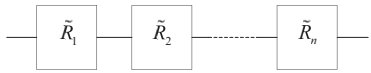


Рис. 5.12 Последовательная система

Нечеткая надежность $\tilde{R}_S = \overset{n}{\underset{i=1}{\otimes}} \tilde{R}_i$ последовательной системы, показанной на рис. 5.12, может быть рассчитана по алгоритму, описанному в разделе 3 для умножения, где $\tilde{R}_i$ представляет надежность i-й компоненты..

Паралельная система. Рассмотрим параллельную систему, которая состоит из "n" компонентов, показанных на рисунке 5.13.

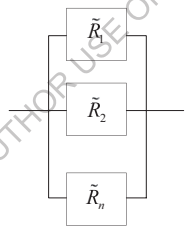


Рис. 5.13 Параллельная система

Нечеткая надежность $\tilde{R}_P = 1 \ominus \overset{n}{\underset{i=1}{\otimes}} (1 \ominus \tilde{R}_i)$ параллельной системы, приведенной на рис. 5.13, может быть рассчитана по алгоритму, приведенному в секции 3, с использованием операций умножения и вычитания, где $\tilde{R}_i$ представляет надежность *i*-й компоненты.

Параллельно-последовательная система. Рассмотрим параллельно-последовательную систему, которая состоит из "m" ветвей, объединенных параллельно. В каждой ветке по 'n' компонентов, как показано на рис. 5.14.

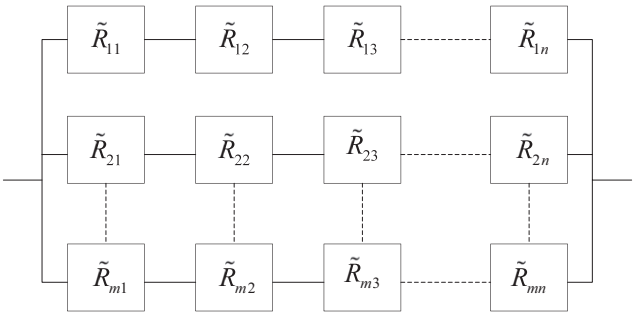


Рис. 5.14 Параллельно-последовательная система

Нечеткая надежность $\tilde{R}_{PS} = 1 \ominus \overset{m}{\underset{k=1}{\otimes}} (1 \ominus (\overset{n}{\underset{i=1}{\otimes}} \tilde{R}_{ki}))$ параллельно-последовательной системы может быть рассчитана по алгоритму, приведенному в секции 3, с помощью операций умножения и вычитания, где $\tilde{R}_{ki}$ представляет надежность *i*-й компоненты (i=1,2,…,n) в k-ой ветке (k=1,2,…,m).

Последовательно-параллельная система. Рассмотрим последовательно-параллельную систему, которая состоит из "n" последовательных подсистем, каждая из которых состоит из 'm' параллельных компонентов (см. рис. 5.15).

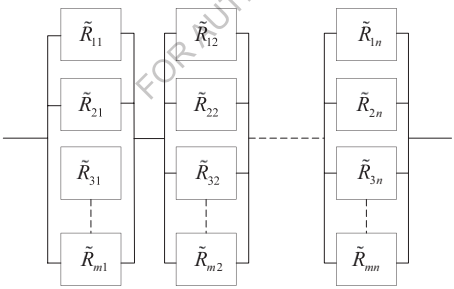


Рис. 5.15 Последовательно-параллельная система

Нечеткая надежность $\tilde{R}_{SP} = \overset{n}{\underset{k=1}{\otimes}} (1 \ominus \overset{m}{\underset{i=1}{\otimes}} (1 \ominus \tilde{R}_{ik}))$ последовательно-параллельной системы может быть рассчитана по алгоритму, описанному в секции 3, с помощью операций умножения и вычитания, где $\tilde{R}_{ik}$ представляют надежность *i*-ой компоненты (i=1,2,…,m) в k-ой подсистеме (k=1,2,…,n).

5.3 Примеры решения задач оценки надёжности нечётких систем без резервирования и с многократным резервом

Для иллюстрации рассматриваемого алгоритма рассчитаем нечеткую надежд ность последовательной системы $\tilde{R}_S$ (рис. 5.16), параллельной системы $\tilde{R}_P$ (рис. 5.17), параллельно-последовательной системы $\tilde{R}_{PS}$ (рис. 5.18) и последовательно-параллельной системы $\tilde{R}_{SP}$ (рис. 5.19), каждая из которых состоит из 4-х компонентов.

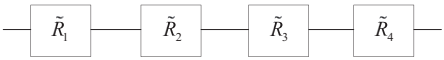


Рис. 5.16 Пример последовательной системы

Последовательная схема соответствует, например, цепи подсистем в иерархической системе управления сортировочной станцией: НО-НА-МККпс-КеОМ.

Пусть надежность компонентов представлена различными типами разговоре-х множеств $\tilde{R}_1, \tilde{R}_2, \tilde{R}_3$ и $\tilde{R}_4$, которые определяются на универсам R, где R представляются размытыми параметрами надежности на интервале [0,1].

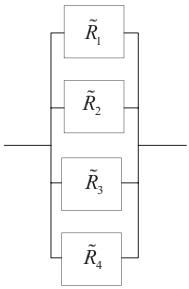


Рис. 5.17 Пример параллельной системы с резервированием

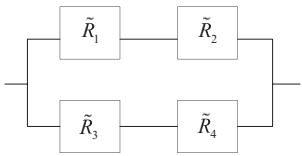


Рис. 5.18 Пример параллельно-последовательной системы

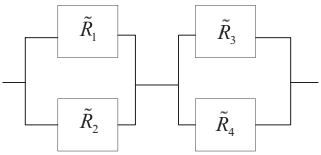


Рис. 5.19 Пример параллельно-последовательной системы

Принятые значения $\tilde{R}_1, \tilde{R}_2, \tilde{R}_3, \tilde{R}_4$ для вычисления $\tilde{R}_S, \tilde{R}_P, \tilde{R}_{PS}, \tilde{R}_{SP}$ сведены в таблицу 5.1 (с унифицированным записи кортежей), а функции принадлежности, описывающие $\tilde{R}_1, \tilde{R}_2, \tilde{R}_3, \tilde{R}_4$, показаны на рис. 5.20.

Таблица 5.1

Нечеткая надежность компонентов

Нечеткая надежность $\tilde{R}_i$ i-й компоненты	Тип размытого множества
$\tilde{R}_1 = <[(0.2, 0.4, 0.4, 0.6); (0.2, 0.4, 0.4, 0.6); 0.7; 0.8]>$	Треугольный
$\tilde{R}_2 = <[(0.3, 0.5, 0.7, 0.9); (0.3, 0.5, 0.7, 0.9); 0.6; 0.9]>$	Трапециевидный
$\tilde{R}_3 = <[(0.6, 0.7, 0.7, 0.8); (0.5, 0.7, 0.7, 0.9); 0.5; 0.7]>$	Треугольный
$\tilde{R}_4 = <[(0.2, 0.3, 0.4, 0.5); (0.1, 0.3, 0.4, 0.6); 0.4; 0.6]>$	Трапециевидный
$\tilde{R}_S = \tilde{R}_1 \otimes \tilde{R}_2 \otimes \tilde{R}_3 \otimes \tilde{R}_4$	
$\tilde{R}_P = 1 \ominus (1 \ominus \tilde{R}_1) \otimes (1 \ominus \tilde{R}_2) \otimes (1 \ominus \tilde{R}_3) \otimes (1 \ominus \tilde{R}_4)$	
$\tilde{R}_{PS} = 1 \ominus (1 \ominus (\tilde{R}_1 \otimes \tilde{R}_2)) \otimes (1 \ominus (\tilde{R}_3 \otimes \tilde{R}_4))$	
$\tilde{R}_{SP} = 1 \ominus (1 \ominus \tilde{R}_1) \otimes (1 \ominus \tilde{R}_2)) \otimes (1 \ominus (1 \ominus \tilde{R}_3) \otimes (1 \ominus \tilde{R}_4))$	

Решение примеров

Шаг 1

Находим значения $\alpha_1, \alpha_2, \alpha_3, \alpha_4, \beta_1, \beta_2, \beta_3$ и β_4 из заданных значений $\tilde{R}_1, \tilde{R}_2, \tilde{R}_3$ и $\tilde{R}_4$:

$\alpha_1 = 0.7,\ \alpha_2 = 0.6,\ \alpha_3 = 0.5,\ \alpha_4 = 0.4;$

$\beta_1 = 0.8,\ \beta_2 = 0.9,\ \beta_3 = 0.7,\ \beta_4 = 0.6.$

Шаг 2

Находим

$\alpha = \min(0.7, 0.6, 0.5, 0.4) = 0.4$ i $\beta = \min(0.8, 0.9, 0.7, 0.6) = 0.6.$

Шаг 3

Вычисляем интервалы надежности для всех компонентов, как точные значения $t_{\tilde{R}_i}(R)$, относящиеся диапазона [0,0.4] (см. таблицу 5.2).

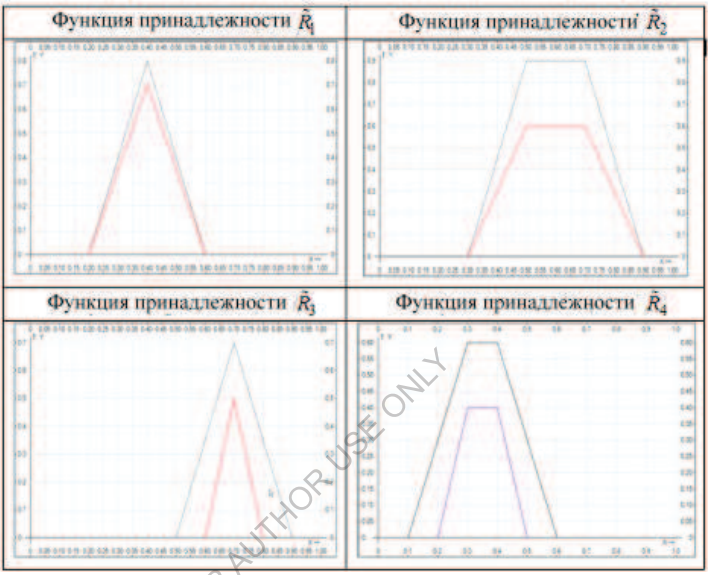


Рис. 5.20 Функции принадлежности для компонентов с нечеткой надежностью

Таблица 5.2

Интервалы надежности для точных значений $t_{\tilde{R}_i}(R)$

Значение $t_{\tilde{R}_i}(R)$	Интервалы надежности $\tilde{R}_1$	Интервалы надежности $\tilde{R}_2$	Интервалы надежности $\tilde{R}_3$	Интервалы надежности $\tilde{R}_4$
0	[0.200,0.600]	[0.300,0.900]	[0.600,0.800]	[0.200,0.500]
0.1	[0.229.0.571]	[0.333,0.867]	[0.620,0.780]	[0.225,0.475]
0.2	[0.257,0.543]	[0.367,0.833]	[0.640,0.760]	[0.250,0.450]
0.3	[0.286,0.514]	[0.400,0.800]	[0.660,0.740]	[0.275,0.425]
0.4	[0.314,0.486]	[0.433,0.767]	[0.680,0.720]	[0.300,0.400]

Шаг 4

Вычисляем интервалы надежности для всех компонентов, как точные значения $1 - f_{\tilde{R}_i}(R)$, относящиеся диапазона [0,0.6] (см. таблицу 5.3).

Таблица 5.3

Интервалы надежности для точных значений $1-f_{\tilde{R}_i}(R)$

Значение $1-f_{\tilde{R}_i}(R)$	Интервалы надежности $\tilde{R}_1$	Интервалы надежности $\tilde{R}_2$	Интервалы надежности $\tilde{R}_3$	Интервалы надежности $\tilde{R}_4$
0	[0.200,0.600]	[0.300,0.900]	[0.500,0.900]	[0.100,0.600]
0.1	[0.225.0.575]	[0.322,0.878]	[0.529,0.871]	[0.133,0.567]
0.2	[0.250,0.550]	[0.344,0.856]	[0.557,0.843]	[0.167,0.533]
0.3	[0.275,0.525]	[0.367,0.833]	[0.586,0.814]	[0.200,0.500]
0.4	[0.300,0.500]	[0.389,0.811]	[0.614,0.786]	[0.233,0.467]
0.5	[0.325,0.475]	[0.411,0.789]	[0.643,0.757]	[0.267,0.433]
0.6	[0.350,0.450]	[0.433,0.767]	[0.671,0.729]	[0.300,0.400]

Полученные функции после выполнения шагов 3 и 4 показаны на рис. 5.21.

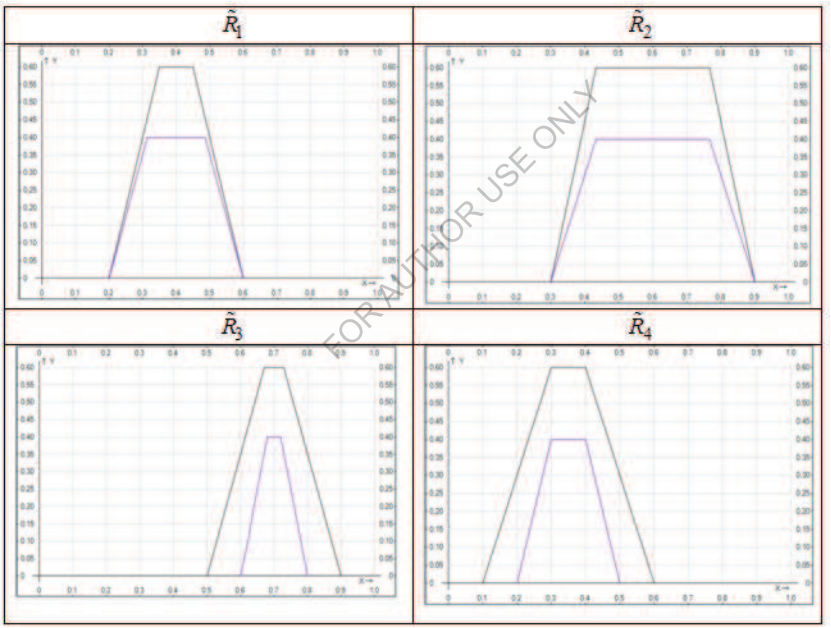


Рис. 5.21 Графики функций с α и β отсечениемням

Шаг 5

Интервалы надежности для последовательной, параллельной, параллельно-последовательной и последовательно-параллельной систем (см. рис. 5.16, 5.17, 5.18, 5.19) можно вычислить с использованием таблиц 5.1, 5.2,

5.3 и алгоритма в секции 3 для точных значений $t_{\tilde{R}_i}(R)$, принадлежащих диапазона [0,0.4]. Полученные результаты приведены в таблице 5.4.

Таблица 5.4

Интервалы надежности для точных значений

Значение $t_{\tilde{R}_i}(R)$	Интервалы надежности $\tilde{R}_S$	Интервалы надежности $\tilde{R}_P$	Интервалы надежности $\tilde{R}_{PS}$	Интервалы надежности $\tilde{R}_{SP}$
0	[0.007,0.216]	[0.821,0.996]	[0.173,0.724]	[0.299,0.864]
0.1	[0.011.0.183]	[0.849,0.993]	[0.205,0.682]	[0.343,0.834]
0.2	[0.015,0.155]	[0.873,0.990]	[0.239,0.640]	[0.387,0.802]
0.3	[0.021,0.129]	[0.894,0.985]	[0.275,0.596]	[0.431,0.768]
0.4	[0.028,0.107]	[0.913,0.980]	[0.312,0.553]	[0.474,0.732]

Шаг 6

Интервалы надежности для последовательной, параллельной, параллельно-последовательной и последовательно-параллельной систем (см. рис. 5.16, 5.17, 5.18, 5.19) можно вычислить с использованием таблиц 5.1, 5.2, 5 .3 и алгоритма в секции 3 для точных значений $1-f_{\tilde{R}_i}(R)$, принадлежащих диапазона [0,0.6]. Полученные результаты приведены в таблице 5.5.

Шаг 7

Функции принадлежности, которые представляются нечеткую надежность последовательной, параллельной, параллельно-последовательной и последовательно-параллельной системы показаны на рис. 5.22.

Таблица 5.5

Интервалы надежности для точных значений $1-f_{\tilde{R}_i}(R)$

Значение $1-f_{\tilde{R}_i}(R)$	Интервалы надежности $\tilde{R}_S$	Интервалы надежности $\tilde{R}_P$	Интервалы надежности $\tilde{R}_{PS}$	Интервалы надежности $\tilde{R}_{SP}$
0	[0.003,0.292]	[0.748,0.998]	[0.107,0.788]	[0.242,0.922]
0.1	[0.005.0.249]	[0.785,0.997]	[0.138,0.749]	[0.281,0.895]
0.2	[0.008,0.212]	[0.818,0.995]	[0.171,0.709]	[0.321,0.867]
0.3	[0.012,0.178]	[0.848,0.993]	[0.206,0.666]	[0.362,0.835]
0.4	[0.017,0.149]	[0.873,0.989]	[0.243,0.624]	[0.403,0.802]
0.5	[0.023,0.123]	[0.896,0.985]	[0.282,0.580]	[0.445,0.767]
0.6	[0.031,0.101]	[0.915,0.979]	[0.322,0.536]	[0.486,0.730]

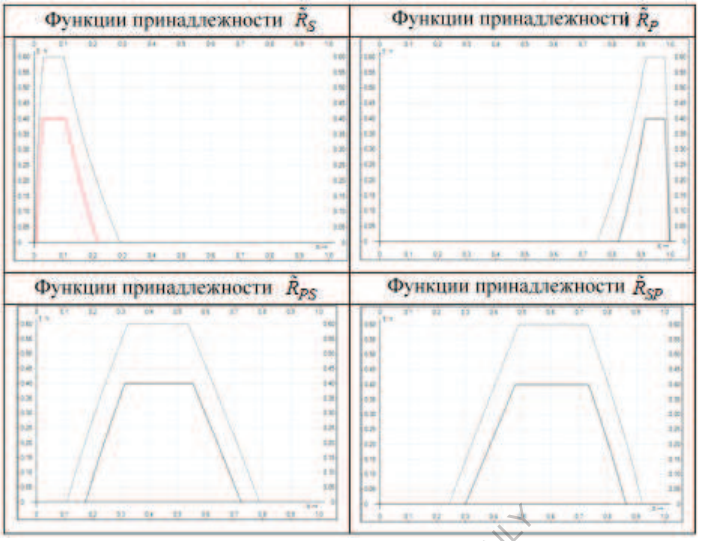


Рис. 5.22 Надежность нечетких систем с размытыми надежностными параметрами компонентов

Далее приведены электронные таблицы с унифицированным описанием компонентов структур и расчетными формулами для обработки исходных данных и оценки показателей надежности нечеткой системы с размытыми границами.

1. Исходные данные компонентов $\tilde{R}_1, \tilde{R}_2, \tilde{R}_3, \tilde{R}_4$.

	B	C	D	E	F	G
3		R1	R2	R3	R4	
4	a1	0,2	0,3	0,6	0,2	
5	b12	0,4	0,5	0,7	0,3	
6	b22	0,4	0,7	0,7	0,4	
7	c1	0,6	0,9	0,8	0,5	
8	a2	0,2	0,3	0,5	0,1	
9	b21	0,4	0,5	0,7	0,3	
10	b22	0,4	0,7	0,7	0,4	
11	c2	0,6	0,9	0,9	0,6	min
12	alfa	0,7	0,6	0,5	0,4	**0,4**
13	betta	0,8	0,9	0,7	0,6	**0,6**

МИН(C12:F12)

2. Обработка исходных данных: построение графиков с α и β отсечением.

	B	C	D	E	F	G	H	I	J	K
14										
15	tR	R1		R2		R3		R4		
16	0	0,200	0,600	0,300	0,900	0,600	0,800	0,200	0,500	
17	0,1	0,229	0,571	0,333	0,867	0,620	0,780	0,225	0,475	
18	0,2	0,257	0,543	0,367	0,833	0,640	0,760	0,250	0,450	
19	0,3	0,286	0,514	0,400	0,800	0,660	0,740	0,275	0,425	
20	0,4	0,314	0,486	0,433	0,767	0,680	0,720	0,300	0,400	
21	**0,5**	**0,343**	**0,457**	**0,467**	**0,733**	**0,700**	**0,700**			$F$7-B17*($F$7-$F$6)/$F$12
22	**0,6**	**0,371**	**0,429**	**0,500**	**0,700**					
23	**0,7**	**0,400**	**0,400**							B17*($F$5-$F$4)/$F$12+$F$4
25	1-fR	R1		R2		R3		R4		
26	0	0,200	0,600	0,300	0,900	0,500	0,900	0,100	0,600	
27	0,1	0,225	0,575	0,322	0,878	0,529	0,871	0,133	0,567	
28	0,2	0,250	0,550	0,344	0,856	0,557	0,843	0,167	0,533	
29	0,3	0,275	0,525	0,367	0,833	0,586	0,814	0,200	0,500	
30	0,4	0,300	0,500	0,389	0,811	0,614	0,786	0,233	0,467	
31	0,5	0,325	0,475	0,411	0,789	0,643	0,757	0,267	0,433	
32	0,6	0,350	0,450	0,433	0,767	0,671	0,729	0,300	0,400	
33	**0,7**	**0,375**	**0,425**	**0,456**	**0,744**	**0,700**	**0,700**			
34	**0,8**	**0,400**	**0,400**	**0,478**	**0,722**					
35	**0,9**			**0,500**	**0,700**					

3. Расчет показателей надежности (по формулам таблицы Н1)

	B	C	D	E	F	G	H	I	J	K
										1-(1-C16)*(1-E16)*(1-G16)*(1-I16) / (1-(1-C16*(1-E16))*(1-(1-G16)*(1-I16))
36										C16*E16*G16*I16 / 1-(1-C16*E16)*(1-G16*I16)
37	tR	Rs		Rp		Rps		Rsp		
38	0	0,007	0,216	0,821	0,996	0,173	0,724	0,299	0,864	
39	0,1	0,011	0,183	0,849	0,993	0,205	0,682	0,343	0,834	
40	0,2	0,015	0,155	0,873	0,990	0,239	0,640	0,387	0,802	
41	0,3	0,021	0,129	0,894	0,985	0,275	**0,597**	0,431	0,768	
42	0,4	0,028	0,107	0,913	0,980	0,312	0,553	0,474	0,732	
44	1-fR	Rs		Rp		Rps		Rsp		
45	0	0,003	0,292	0,748	0,998	0,107	0,788	0,242	0,922	
46	0,1	0,005	0,249	0,785	0,997	0,138	0,749	0,281	0,895	
47	0,2	0,008	0,212	0,819	0,995	0,171	0,709	0,321	**0,866**	
48	0,3	0,012	0,178	0,848	0,993	0,206	**0,667**	0,362	0,835	
49	0,4	0,017	0,149	0,874	0,989	0,243	0,624	0,403	0,802	
50	0,5	0,023	0,123	0,896	0,985	0,282	0,580	0,445	0,767	
51	0,6	0,031	0,101	0,915	0,979	0,323	0,536	0,486	0,730	

СПИСОК ЛИТЕРАТУРЫ

1. Автоматизированная система организационно-технологического управления сортировочными станциями (АСОТУ СС). Разработка технического задания на ситему в целом [Текст] : Звіт про НДР. Т. 99.01.85.87 / Дніпропетровський інститут інженерів залізничного транспорту; керівн. А.А. Косолапов; викон. : А.А. Косолапов. — Дніпропетровськ : ДІІТ, 1986. — 95 с. - держреєстрація № 01860063785.

2. Всероссийская практическая конференция «Стандарты в проектах современных информационных систем» [Электронный ресурс] / Режим доступа : http://www.osp.ru/cw/2003/20/64757/ // Computerworld Россия. 2003. № 20.

3. ГОСТ 24.701-83 Автоматизированные системы управления технологическими процессами. Надежность. Основные понятия. [Текст]. — М : Издательство стандартов, 1983. — 17 с.

4. ГОСТ 34.003-90 [Текст] // Информационная технология. Комплекс стандартов на автоматизированные системы. Автоматизированные системы. Термины и определения. 1990.

5. ГОСТ 23501.108-85 [Текст] // Системы автоматизированного проектирования. Классификация и обозначение. 1985.

6. Замедлитель вагонный РНЗ-2М [Електронний ресурс] / Режим доступу : http://www.zrmz.ru/catalog/24/88/ // Златоустовский ремонтно-механический завод. 2013.

7. Информационная технология. Комплекс стандартов и руководящих документов на автоматизированные системы (ГОСТ 34.201-89, ГОСТ 34.602-89, РД-50-682-89, РД-50-680-88, ГОСТ 34.601-90, ГОСТ 34.401-90, РД-50-698-90, ГОСТ 34.003-90, РД-50-34.119-90) [Текст] —М: Госстандарт СССР, 1991. — 143 с.

8. Концепція автоматизації роботи сортувальних гірок України [Текст] / Проект. : ДП ДНДЦУЗ, 2010. — 36 с.

9. НІРО-технология. Общее описание. [Текст]. — М : ПКТБ АСУЖТ, 1980. — 48 с.

10. Науково-дослідна та дослідно-конструкторська робота [Электронный ресурс] // Режим доступа: URL: http://www.youtube.com/watch?v=3gjrlBp9rvc&feature=relmfu/. 2013.

11. "НТЦ ТРАНГССИСТЕМОТЕХНИКА": на высоте сегодняшних требований [Текст] // Евразия Вести. 2011. № II.

12. Общеотраслевые руководящие методические материалы по созданию и применению автоматизированных систем управления технологическими процессами в отраслях промышленности (ОРММ-3 АСУТП) [Текст] / Государственный комитет СССР по науке и технике. — М : Финансы и статистика, 1986. — 73 р.

13. Про схвалення Транспортної стратегії України на період до 2020 року // Розпорядження Кабінету Міністрів України від 20 жовтня 2010 р. № 2174-р, Київ. 2010.

14. Рекомендации по механизации и автоматизации сортировки вагонов на горках. Р 835 [Текст] // III издание. Разработано экспертами Комиссии ОСЖД по инфраструктуре и подвижному составу 25-27 мая 2004 г. — Варшава, 2004. — 11 с.

15. Техническое задание на создание автоматизированной системы управления маршрутами движения отцепов на сортировочной горке ГАЦ (АСУМД) [Текст] —М: МПС, 1981. — 40 с.

16. "Укрзализныця" запустила Единый центр обработки данных / Интерфакс-Украина, 26.07.2012. // [Электронный ресурс]. - Режим доступа: URL: http://interfax.com.ua/news/economic/112240.html 2012.

17. Ададуров С.Е. Интеллектуальный железнодорожный транспорт [Текст] // Автоматика, связь, информатика. 2011. № 6. — С. 4-8.

18. Алиев Т.И. Основы моделирования дискретных систем [Текст] / Т.И. Алиев. — Санкт-Петербург : СПбГУ ИТМО, 2009. — 363 с.

19. Архангельский Е.В. Надёжность технических устройств и её влияние на перерабатывающую способность станций [Текст] / Е.В. Архангельский, В.П. Шейкин // Вестник ВНИИЖТ. 1979. № 7. — С. 1-5.

20. АС № 45271 від 21.08.2012, Україна, ДДІВ. Комп'ютерна програма "Генетичний алгоритм оптимізації структур інформаційних систем (GAOSIS)" / Косолапов А.А., Гриненко Ю.О. . 2012.

21. АС № 45855 від 02.10.2012, Україна, ДДІВ. Комп'ютерна програма "Вибір пріоритетів потоків заявок в інформаційно-керуючих системах реального часу» («PRIORY»)" / Косолапов А.А., Борисов О.І., 2012.

22. АС № 45857 від 02.10.2012, Україна, ДДІВ. Комп'ютерна програма "Оптимізація інформаційно-керуючих обчислювальних систем» («ОПТІКОС»)" / Косолапов А.А., Мудрик О.Б., 2012.

23. АС № 45858 від 02.10.2012, Україна, ДДІВ. Комп'ютерна програма "Комплекс системного інтегратора» («CSI»)" / Косолапов А.А., Магамедов В.К.,

2012.

24. АС № 1073146, МКИ, В61, L17/00. Устройство для управления технологическим процессом роспуска составов на сортировочной горке / Г.А. Красовский, Л.В. Сафрис, А.А. Косолапов, Б.М. Филимонов, Е.М. Шафит, Е.В. Щербаков. - № 3294504/27-11; заявлено 03.04.81; опубл. 15.02.84, бюл. №6. — СССР, 1984. — 2 с..

25. Асанов М.О. Дискретная математика: графы, матроиды, алгоритмы [Текст] / М.О. Асанов, В.А. Барановский, В.В. Расин —Ижевск : НИЦ "Регулярная и хаотическая динамика", 2001. — 288 р.

26. Байцер Б. Архитектура вычислительных систем [Текст] / Б. Байцер. Т. 1 — Москва : МИР, 1974. — 498 р.

27. Беляев Ю.К. Надежность технических систем : Справочник. Под. ред. проф. И.А. Ушакова [Текст] / Ю.К. Беляев. — М : Радио и связь, 1985. — 608 с.

28. Бенсусан А. Методы декомпозиции, децентрализа-ции, координации и их приложения [Текст] / А. Бенсусан, Ж.Л. Лионс, Р. Темам // Методы вычислительной математики — Новосибирск : Наука, 1975.

29. Берндт Т. Сортировочные горки на железных дорогах мира [Текст] / Т. Берндт, С. В. Власенко // Автоматика, связь, информатика. 2007. № 6. — С. 45-48.

30. Бобровский В.И. Оптимизация режимов торможения отцепов на сортировочных горках : Монография / В.И. Бобровский и др. — Дн-вск : Изд-во Маковецкий, 2010. — 260 с.

31. Бондарев В.М. Основы программирования [Текст] / В.М. Бондарев, В.И. Рублинецкий, Е.Г. Качко. — Харьков : Фолио, 1998. — 256 с.

32. Босов А.А. Структурная сложность систем [Текст] / А. А. Босов , В. М. Ильман // Вісник ДНУЗТ ім. ак. В. Лазаряна. 2012. № 40. — С. 173-179.

33. Бочаров А.П. Разработка и внедрение корпоративной информационной системы: основные положения и проблемы [Текст] / А. П. Бочаров, П. П. Науменко, Ф. А. Карбинский, В. А. Шиш // Інформаційно-керуючі системи на залізничному транспорті. 2012. № 5. — С. 3-8.

34. Бочаров О.П. Динамічна модель сортувальної станції та її роль в подальшій оптимізації процесу перевезень [Текст] / О. П. Бочаров, Г. О. Міхальов, В. П. Мороз, В. О. Шиш // Інформаційно-керуючі системи на залізничному транспорті. 2011. № 5. — С. 74-76.

35. Буянов В.А. Автоматизированные информационные системы на

железнодорожном транспорте [Текст] / В.А. Буянов, Г.С. Ратин. — М : Транспорт, 1984. — 239 с.

36. Вагин В.Н. Достоверный и правдоподобный вывод в интеллектуальных системах [Текст] / В. Н. Вагин, Е. Ю. Головина, А.А. Загорянская, М. В. Фомина —М. : ФИЗМАТЛИТ, 2004. — 704 с.

37. Вальков В.М. Микроэлектронные управляющие вычислительные комплексы. Системное проектирование и конструирование [Текст] / В.М. Вальков. — Л : Машиностроение. Ленингр. отд-ние, 1990. — 224 с.

38. Великодный В.В. Задачи по эксплуатации вагонных парков на основе ав-томатизированной системы управления грузовыми перевозками Укрзализныци [Текст] / В. В. Великодный, В. Б. Землянов, В. В. Скалозуб, И. В. Жуковицкий, С. Ю. Цейтлин // Інформаційно-керуючі системи на залізничному транспорті. 2005. № 3. — С. 31-35.

39. Гельфанд И.М. О некоторых способах управления сложными системами [Текст] / И. М. Гельфанд, М.Л. Цетлин // Успехи математических наук. 1962. Т. 17. № 1 (103). — С. 3-25.

40. Гильман А.С. Учет характеристик программного обеспечения при анализе надежности УВК АСУТП на этапе проектирования [Текст] / А.С. Гильман // Приборы и системы управления. 1982. № 3. — С. 8-11.

41. Де Гроот М. Оптимальные статистические решения [Текст] / М. Де Гроот. — М. : Мир, 1974.

42. Древс Ю.Г. Системы реального времени: технические и программные средства [Текст] / Ю.Г. Древс. — М : МИФИ, 2010. — 320 с.

43. Душкин Д.Н. Сетецентрические технологии: эволюция, текущее положение и области дальнейших исследований [Текст] / Д. Н. Душкин, М. П. Фархадов // Автоматизация и современные технологии. - М.: Машиностроение. 2012. № 1. — С. 21-29.

44. Жуковицький І.В. Розвиток теорії та удосконалення систем автоматичного управління швидкістю скочування відчепів на сортувальних гірках [Текст] : автореф. дис. .... д.т.н. 05.22.08 / Жуковицький Ігор Володимирович. — Харків : Харківська державна академія залізничного транспорту, 1999. — 34 с.

45. Забегалин Е.В. Архитектура информационных систем в теории и практике. IBS. Департамент управленческого консалтинга [Электронный ресурс] / Е. В. Забегалин // Режим доступа : http://www.evz.name/evzms-2.pdf. 2012.

46. Зиглер К. Методы проектирования программных систем, Пер. с англ. [Текст] / К. Зиглер. — М : Мир, 1985. — 328 с.

47. Иванченко В.И. Новый подход к управлению процессом роспуска составов на сортировочной горке [Текст] / В. И. Иванченко, Н. Н. Лябах, А. А. Сепетый // Труды РИИЖТа. - Ростов-на-Дону. 1984. — С. 34-41.

48. Иванченко В.М. Микропроцессорные информационно-управляющие системы автоматизации сортировочных процессов. Уч. пособие [Текст] / В. М. Иванченко. — Ростов-на-Дону, 1984. — 96 с.

49. Иванченко В.Н. Теория построения и реализация информационно-управляющих микропроцессорных систем на сортировочных станциях [Текст] : автореф. дис. .... д.т.н. 05.22.08 / Иванченко Владимир Николаевич. — Ленинград : ЛИИЖТ имени академика В. Н. Образцова, 1988. — 48 с.

50. Иванюто И.Д. АСУ сортировочными станциями (на примере АСУ СС НПО «Агат») [Текст] / И. Д. Иванюто, С. С. Галуза, А. А. Ерофеев, Н. Н. Казаков —Гомель : Белорусский государственный университет транспорта, 2003. — 159 с.

51. Ишков П.В. Виртуализация: основа построения эффективных ИТ [Текст] / П. В. Ишков // Автоматика, связь, информатика. 2012. № 2. — С. 11-13.

52. Івченко Ю.М. Інтеграція мережевого обладнання АСК ВП УЗ та АСК ПП УЗ, підключення його до ЄМПД [Текст] / Ю. М. Івченко, В. Г. Івченко, О. М. Гондар // Вісник ДНУЗТ ім. ак. В. Лазаряна // 2009. № 29. — С. 143-146.

53. Казиев Г.Д. Задачи технического перевооружения сортировочных станций [Текст] / Г. Д. Казиев, А. Г. Савицкий // Автоматика, связь, информатика. 2007. № 4. — С. 17-22.

54. Каменнова М.С. Системный подход к проектированию сложных систем [Текст] / М.С. Каменнова // Журнал д-ра Добба. 1993. № 1. — С. 9-14.

55. Кинаш С.А. Унификация архитектурных решений для АСУ РЖД [Текст] / С. А. Кинаш // Автоматика, связь, информатика. 2007. № 7. — С. 20-21.

56. Кириллов В.П. SSADM - передовая технология разработки автоматизированных систем [Текст] / В.П. Кириллов // Компьютеры + Программы. 1994. № 2(10). — С. 8-16.

57. Клир Д. Системология. Автоматизация решения системных задач : Пер. с англ. / Дж. Клир. — М : Радио и связь, 1990. — 544 с.

58. Козлов Б.А. Справочник по расчету надежности аппаратуры

радиоэлектроники и автоматики [Текст] / Б.А. Козлов, И.А. Ушаков. — М : Сов.радио, 1975. — 472 с.

59. Косолапов А.А. Автоматизация инженерного проектирования информационно-управляющих вычислительных систем и сетей: методика, модели, методы, программная среда [Текст] / А.А. Косолапов. — Винница : ВПИ, ИК им. ак. В.М.Глушкова, АН Украины, 1993. — С. 114.

60. Косолапов А.А. Автоматизация системного проектирования информационно-управляющих вычислительных систем и сетей АСУТП [Текст] / А.А. Косолапов // Комп'ютерне моделювання. Дніпродзержинськ : Мін. освіти, ДДТУ 1998. — С. 96-97.

61. Косолапов А.А. Автоматизированная система управления маршрутами движения для сортировочных горок на базе микропроцессорной ЭВМ типа СМ1800 [Текст] / Г. А. Красовский, Б. А. Игнатов, Л. В. Сафрис, Б. М. Филимонов, А. А. Косолапов // Автоматика, телемеханика и связь. 1984. № 7. — С. 7-11.

62. Косолапов А.А. Анализ состояния и перспектив развития средств технической диагностики управляющих вычислительных комплексов в АСУТП [Текст] / А.А. Косолапов, В.Г. Савченко // Автоматизированные информационные системы на сортировочных станциях железнодорожного транспорта. - Межвуз. сб. научн. трудов. - Днепропетровск: изд. ДИИТа. 1988. № 263/16. — С. 5-12.

63. Косолапов А.А. Внедрение информационных технологий для обеспечения качества подготовки специалистов. ЦИТ: 212-725 [Текст] / Б. Е. Боднарь, Е. Б. Боднарь, А. А. Косолапов // Сб. научных трудов SWorld "Перспективные инновации в науке, образовании, производстве и транспорте '2012". Технические науки. - Одесса. 2012. Т. 5. № 2. — С. 31-36.

64. Косолапов А.А. Выбор технических средств и разработка вариантов технической структуры СКАТ-СС [Текст] / Е. М. Шафит, И. В. Жуковицкий, А. А. Косолапов, Т. В. Туник // Інформ.-керуючі системи на залізн. трансп. 2000. № 2. — С. 14-19.

65. Косолапов А.А. Дослідження умов доцільності децентралізації функцій керування в системах гіркової автоматики [Текст] / А.А. Косолапов. ЦИТ: 213-903 // Сборник научных трудов SWorld. 2013. Т. 2. № 2. — С. 37-45.

66. Косолапов А.А. Информационно-методическая база управления учебным процессом в техническом университете [Текст] / А. Н. Пшинько, А. С. Распопов, А. А. Косолапов // Информационная инфраструктура высших

учебных заведений. - Сборник научных трудов. 1999. Т. 2. — С. 85-88.

67. Косолапов А.А. Исследование механизмов защиты систем в условиях сбоев и деградации с использованием иерархических сетей Петри [Текст] / А. А. Косолапов, М. А. Демчук // Вісник Технологічного університету Поділля. Технічні науки. Науковий журнал, Хмельницький. 2003. Т. 2. № 3. — С. 64-67.

68. Косолапов А.А. К вопросу структурного проектирования автоматизированных систем [Текст] / А. А. Косолапов, М. А. Косолапова // Математичне моделювання. Науковий журнал. Дніпродзержинськ: - Мін. освіти, ДДТУ. 2000. № № 2(5). — С. 124-128.

69. Косолапов А.А. Ключевая роль транспорта в современном мире : монография [Текст] / [авт. кол. : Косолапов А. А., Блохин А. Л., Боряк К. Ф. и др.]. — Одесса : КУПРИЕНКО СВ, 2013. — 163 с. - ISBN 978-966-2769-16-6.

70. Косолапов А.А. Концептуальні моделі гіркових процесів і систем. ЦИТ: 412-0950. [Текст] / А. А. Косолапов // Сборник научных трудов SWorld. Материалы международной научно-практической конференции «Современные проблемы и пути их решения в нау-ке, транспорте, производстве и образовании'2012». - Технические науки. Информатика, вычислительная техника и автоматизация. - Одесса: КУПРИЕНКО. 2012. Т. 14. № 4. — С. 71-81.

71. Косолапов А.А. Концептуальні моделі сортувальних станцій. ЦИТ: 412-0949. [Текст] / А. А. Косолапов // Сборник научных трудов SWorld. Материалы международной научно-практической конференции «Современные проблемы и пути их решения в науке, транспорте, производстве и образовании'2012». - Транспорт. Техническая эксплуатация и ремонт средств транспорта. - Одесса: КУПРИЕНКО. 2012. Т. 2. № 4. — С. 65-69.

72. Косолапов А.А. Концепція побудови, розробки і впровадження автоматизованої інформаційної системи управління університетом на базі мережевих інформаційних інтернет-технологій [Текст] / О. М. Пшінько, В. Я. Нечай, Є.М.Шафіт, А.А. Косолапов // Інформ.-керуючі системи на залізн. трансп. 1998. № 4. — С. 70-71.

73. Косолапов А.А. Логіко-лінгвістична система управління сповільнювачем прицільної гальмівної позиції на сортувальній гірці. ЦИТ: 212-750 [Текст] / А. А. Косолапов // Сб. научных трудов SWorld. "Перспективные инновации в науке, образовании, производстве и транспорте '2012 ". - Транспорт. Физика и математика. - Одесса. 2012. Т. 2. № 2. — С. 58-61.

74. Косолапов А.А. Машинная методика эскизной компоновки и расчета характеристик децентрализованных УВК систем горочной автоматики на

МКЭВМ [Текст] / А.А. Косолапов, О.В. Рожков // Межвуз. сб. научн. трудов. - Днепропетровск: изд ДИИТа. 1984. № 236/13. — С. 24-29.

75. Косолапов А.А. Методика анализа надёжности нечётких систем с использованием теории размытых множеств. ЦИТ: 113-1008. [Текст] / А. А. Косолапов // Сборник научных трудов SWorld. Материалы международной научно-практической конференции «Современные направления теоретических и прикладных исследований ‘2013». - Технические науки. Информатика, вычислительная техника и автоматизация. - Одесса: КУПРИЕНКО. 2013. Т. 10. № 1. — С. 24-35.

76. Косолапов А.А. Методика выбора технических структур управляющих систем сортировочных горок по критерию эффективного использования вычислительных ресурсов [Текст] / А.А. Косолапов // (в печати). 2013.

77. Косолапов А.А. Методика разработки, структура и характеристика микропроцессорной информационно-управляющей вычислительной системы для ГАЦ (АСУ МД) : сборник научных трудов [Текст] / Е. М. Шафит, Л. В. Сафрис, А. А. Косолапов // Автоматизация управления маршрутами на сортировочных горках : Сб. науч. тр. ВНИИЖТ. М. 1984. — С. 70-78.

78. Косолапов А.А. Методология автоматизированного системного анализа и проектирования как основа создания информационно-управляющих вычислительных систем и сетей / А. А. Косолапов // Информационные технологии на железнодорожном транспорте “ИНФОТРАНС-96”. — СПб. : ПГУПС, 1996. — С. 332-341.

79. Косолапов А.А. Методологія проектування інтегрованих комп'ютерних систем управління організаційно-технологічними процесами на залізничному транспорті [Текст] / А. А. Косолапов // Інформ.-керуючі системи на залізн. трансп. . 1998. № 4. — С. 69-70.

80. Косолапов А.А. Многоуровневая структурная оптимизация в составе инженерной методики проектирования корпоративных информационных систем [Текст] / А. А. Косолапов // Математичне моделювання. Науковий журнал. Дніпродзержинськ: - Мін. освіти, ДДТУ. 2000. № 1(4). — С. 57-60.

81. Косолапов А.А. Многоуровневая структурная оптимизация в составе инженерной методики проектирования корпоративных информационных систем [Текст] / А. А. Косолапов // Автоматика-2000. Міжнародна конференція з автоматичного управління, Львів, 11-15 вересня 2000: Праці в 7-ми томах.- Львів: Державний НДІ інформаційної інфраструктури. 2000. Т. 6. — С. 274-278.

82. Косолапов А.А. Моделі дискретних систем реального масштабу часу керування сортувальними гірками. ЦИТ: 312-793 [Текст] / А. А. Косолапов // Сб. научных трудов SWorld. «Научные исследования и их практическое применение. Современное состояние и пути развития `2012». - Транспорт. Физика и математика. - Одесса. 2012. Т. 2. — С. 46-58.

83. Косолапов А.А. Некоторые характеристики микропроцессоров и микро-ЭВМ и технологических алгоритмов в системах управления роспуском составов на горке [Текст] / Е.М. Шафит, Л.В. Сафрис, А.А. Косолапов // Труды. Межвузовский тематический сборник. Ростов-на-Дону. РИИЖТ. 1982. № 168. — С. 20-23.

84. Косолапов А.А., Пшінько Ю.О. Онтологічні моделі в задачах автоматизації сортувальних станцій [Текст] / А.А. Косолапов, Ю.О. Пшінько // Сборник научных трудов SWorld. Материалы международной научно-практической конференции «Научные исследования и их практическое применение. Современное состояние и пути развития ‘2013». – Выпуск 4. Том 12. Транспорт. Техническая эксплуатация и ремонт средств транспорта. – Одесса: КУПРИЕНКО, 2013. – ЦИТ: 410-0947. – С. 65-69.

85. Косолапов А.А. Оптимизация распределения информационных потоков при проектировании автоматизированных систем / А. А. Косолапов // Оптимизация производственных процессов. - Сб. научн. трудов. Севастополь: изд. СевГТУ. 2000. № 3. — С. 143-146.

86. Косолапов А.А. Организация антивирусной защиты и самодиагностики в распределённых информационных системах [Текст] / А. А. Косолапов, Д. В. Лоскутов, Н. А. Фастов // Інформаційні технології в наукових дослідженнях і навчальному процесі : матеріали V Міжнар. наук.-практ. конф. (17-19 листоп. 2010 р.). - Луганськ. 2010. — С. 176-181.

87. Косолапов А.А. Организация подготовки специалистов в области современных сетевых информационных технологий для железных дорог Украины [Текст] : материалы временных коллективов / В. А. Домашний, В. С. Юденко, Е. М. Шафит, А. А. Косолапов. — СПб, 1996. — С. 259-263.

88. Косолапов А.А. Организация управления учебным процессом на основе информационной системы университета [Текст] / А. Н. Пшинько, А. С. Распопов, А. А. Косолапов // 3 Міжнародна науково-методична конференція "Освіта та Віртуальність". Наукові праці. — Харків : ХТУРЕ, 1999. — С. 68-70.

89. Косолапов А.А. Основные направления и этапы развития систем управления сортировочным процессом на станциях [Текст] / Е.М. Шафит, Л.В.

Сафрис, А.А. Косолапов // Автоматизированные системы управления технологическими процессами на сортировочных станциях магистрального и промышленного транспорта. - Межвуз. сб. научн. трудов. - Днепропетровск: изд. ДИИТа. 1985. № 244/14. — С. 3-16.

90. Косолапов А.А. Оценка надежности автоматизированной системы управления маршрутами движения на горке (АСУМД) [Текст] / Л.А. Вашурин, В.Н. Рыбцов, С.Г. Чижов, А.А. Косолапов // Автоматизированные системы управления технологическими процессами на сортировочных станциях магистрального и промышленного транспорта. - Межвуз. сб. научн. трудов. - Днепропетровск: изд. ДИИТа. 1985. № 244/14. — С. 52-62.

91. Косолапов А.А. Подсистема диагностики и обслуживания многомикромашинных информационно-управляющих вычислительных систем автоматизации сортировочных станций [Текст] / А.А. Косолапов, Л.Г. Богдан, А.А. Гарибян // Микропроцессорные системы и устройства управления ответственными технологическими процессами на транспорте. - М.: ЦП ВНТО ж.д. и трансп. строителей, МИИТ. 1989. — С. 45-47.

92. Косолапов А.А. Подсистема контроля, диагностики и обслуживания комплекса технических средств ИПС СС [Текст] / Е.М. Шафит, А.А. Косолапов, Л.Г. Богдан, Н.И. Бубело // Межвуз. сб. научн. трудов. - Днепропетровск: изд. ДИИТа. 1990. № 279/17. — С. 5-10.

93. Косолапов А.А. Подход к оценке влияния диагностики вычислительной системы на показатели ее надежности [Текст] / А.А. Косолапов, В.Г. Савченко // Автоматизированные информационные системы на сортировочных станциях железнодорожного транспорта. - Межвуз. сб. научн. трудов. - Днепропетровск: изд. ДИИТа. 1988. № 263/16. — С. 13-22.

94. Косолапов А.А. Пошук оптимальної структури інформаційної системи. Генетичний підхід. ЦИТ: 113-1007. [Текст] / А. А. Косолапов, Ю. О. Гриненко // Сборник научных трудов SWorld. Материалы международной научно-практической конференции «Современные направления теоретических и прикладных исследований ‘2013». - Транспорт. Транспортные и логистические системы. - Одесса: КУПРИЕНКО. 2013. Т. 1. № 1. — С. 35-44.

95. Косолапов А.А. Применение микропроцессорных средств в автоматизированных системах управления сортировочным процессом на горке [Текст] / Е.М. Шафит, Л.В. Сафрис, А.А. Косолапов // Автоматизированные системы управления технологическими процессами на железнодорожных станциях : Межвуз. сб .науч. тр. - Днепропетровск : ДИИТ. 1982. № 224/11. —

С. 3-12.

96. Косолапов А.А. Принципы декомпозиции и интеграции автоматизированных систем управления сортировочными станциями [Текст] / Е.М. Шафит, А.А. Косолапов // Автоматизированные информационно-управляющие вычислительные системы на сортировочных станциях железнодорожного транспорта. - Межвуз. сб. научн. трудов. - Днепропетровск: изд. ДИИТа. 1993. № 289/18. — С. 5-17.

97. Косолапов А.А. Принципы нечёткой маршрутизации. ЦИТ: 312-792 [Текст] / А. А. Косолапов // Сб. научных трудов SWorld. «Научные исследования и их практическое применение. Современное состояние и пути развития `2012». - Технические науки. - Одесса. 2012. Т. 6. — С. 18-22.

98. Косолапов А.А. Принципы описания общего алгоритма функционирования децентрализованных систем управления технологическими процессами [Текст] / Е.М. Шафит, Л.В. Сафрис, А.А. Косолапов, Ю.А. Косорига // Автоматизированные системы управления технологическими процессами на железнодорожных станциях. Межвуз. сб. науч. тр. - Днепропетровск : ДИИТ. 1983. — С. 28-36.

99. Косолапов А.А. Принципы построения интегрированной автоматизированной системы управления технологическими процессами на сортировочной станции / Е. М. Шафит, И. В. Жуковицкий, А. А, Косолапов // Информационно-управляющие системы на железнодорожном транспорте. 1996. № 1,2. — С. 36-41.

100. Косолапов А.А. Принципы проектирования и структура корпоративной информационной системы университета / А. А. Косолапов // 3 Міжнародна науково-методична конференція "Освіта та Віртуальність". Наукові праці - 104 с. // ХТУРЕ — Харків, 1999. — С. 64-67.

101. Косолапов А.А. Проектування інтелектуальних транспортних систем і мереж засобами аналітичних моделей масового обслуговування [Текст]: Монографія / А. А. Косолапов. — Дніпропетровськ : Вид-во Дніпропетр. нац. ун-ту залізн. трансп. ім. акад. В. Лазаряна, 2013. — 190 с.

102. Косолапов А.А. Развитие технической структуры автоматизированных систем управления технологическими процессами на сортировочных станциях. Система СКАТ-СС / Е. М. Шафит, И. В. Жуковицкий, А. А. Косолапов // Proceedings JУЖЕЛ 7th International Conference of Railway Experts. Yugoslavia, Vrnjacka Banja, October 04-06. 2000. — С. 145-146.

103. Косолапов А.А. Развитие технической структуры управляющих

вычислительных комплексов АСУ МД на базе микроЭВМ СМ1810 : сборник [Текст] / Е.М. Шафит, А.А. Косолапов // Развитие АСУ маршрутами движения на сортировочных горках. - М. 1987. — С. 65-67.

104. Косолапов А.А. Разработка и опыт применения средств автоматизации проектирования локальных вычислительных систем АСУ [Текст] / Е.М. Шафит, А.А. Косолапов // Практическое применение современных технологий программирования, пакетов прикладных программ в вычислительных системах и сетях ЭВМ. Инф. сборник “Приборы, средства автоматизации и системы управления“. Часть 2. - Серия “Научно-техническая пропаганда“. - М.: Информприбор. 1988. № 8. — С. 5-7.

105. Косолапов А.А. Разработка и применение методов математического моделирования при анализе и проектировании микропроцессорных АСУ технологическими процессами роспуска составов на горках [Текст] : автореф. дис. ... к.т.н. 05.13.07 / Косолапов Анатолий Аркадьевич —Москва : Всесоюзный ордена Трудового Красного Знамени научно-исследовательский институт железнодорожного транспорта (ЦНИИ МПС), 1984. — 24 с.

106. Косолапов А.А. Разработка подсистемы диагностики микрокомпьютерных локальных вычислительных сетей АСУ [Текст] / А. А. Косолапов, Л. Г. Богдан, В. Н. Саква, А. П. Тихонов, Н. И. Бубело // Практическое применение современных технологий программирования, пакетов прикладных программ в вычислительных системах и сетях ЭВМ. - Инф. сборник “Приборы, средства автоматизации и системы управления“. Часть 2. - Серия “Научно-техническая пропаганда“. - М.: Информприбор. 1988. № 8. — С. 30-31.

107. Косолапов А.А. Разработка УВК на базе микро-ЭВМ для совершенствования систем горочной автоматики и сопряжение их с АСУСС [Текст] / А.Н. Вахнин, Т.Н. Зиброва, А.А. Косолапов // Автоматизированные системы управления технологическими процессами на железнодорожных станциях: Межвуз. сб. науч. тр. - Днепропетровск : ДИИТ. 1983. № 224/11. — С. 3-12.

108. Косолапов А.А. Разработка унифицированной системы автоматизированного управления процессом расформирования поездов на сортировочной станции [Текст]: материалы временных коллективов / Е. М. Шафит, И. В. Жуковицкий, А. А. Косолапов. — Санкт-Петербург,, 1996. — С. 81-93.

109. Косолапов А.А. Резервы архитектуры автоматизированной системы

управления грузовыми перевозками Украинских железных дорог [Текст] / А. А. Косолапов, И. В. Жуковицкий // Залізничний транспорт України. - Київ, 2013, № 1. — С. 10-13.

110. Косолапов А.А. Системи штучного інтелекту [Текст]: методичні вказівки до практичних і лабораторних робіт із розділу «Нечітке виведення в інтелектуальних системах проектування та управління» / уклад. А. А. Косолапов. — Дніпропетровськ : Вид-во Дніпропетр. нац. ун-ту залізн. трансп. ім. акад. В. Лазаряна, 2012. — 47 с.

111. Косолапов А.А. Системные принципы построения технической структуры автоматизированной информационной системы университета [Текст] / А. А. Косолапов // Информационная инфраструктура высших учебных заведений. - Сборник научных трудов. -Санкт-Петербург. - 98 с. 1999. Т. 2. — С. 48-50.

112. Косолапов А.А. Системотехнический подход к построению структуры интегрированных автоматизированных систем управления сортировочными станциями [Текст] / А.А. Косолапов, Е.М. Шафит // Микропроцессорные системы и устройства управления ответственными технологическими процессами на транспорте. - М.: ЦП ВНТО ж.д. и трансп. строителей, МИИТ 1989. — С. 42-44.

113. Косолапов А.А. Таблично функциональное моделирование в задачах проектирования управляющих вычислительных систем автоматизации сортировочных станций [Текст] / А.А. Косолапов // Некоторые вопросы математического моделирования в инженерных задачах. - Сб. научн. трудов. - Днепропетровск: изд. ДИИТа. 1994. № 297/3. — С. 66-74.

114. Косолапов А.А. Тенденції розвитку архітектури автоматизованих систем керування [Текст] / А. А. Косолапов, І. В. Жуковицький // Системные технологии. Региональный межвузовский сборник научных работ. 2013. № 3 (86). — С. 62-71.

115. Косолапов А.А. Формирование рациональной структуры телекоммуникационной среды корпорации на основе эволюционно-генетического подхода [Текст] / А. А. Косолапов, А. Б. Мудрик // Информационные технологии на железнодо-рожном транспорте. Доклады десятой международной научно-практической конференции „Инфотранс-2005”. - СПб. 2005. — С. 135-140.

116. Косолапов А.А., Рыбцов В.Н. Оценка эффективности функционирования отдельных вариантов структуры микропроцессорных АСУ

роспуском составов на горке с учётом надёжности вычислительного комплекса [Текст] / А.А. Косолапов, В.Н. Рыбцов // В. сб.: Автоматизированные системы управления технологическими процессами на железнодорожных станциях. - Днепропетровск: ДИИТ. 1982. — С. 81-91.

117. Крицкий С.П. Трансляция языков программирования: синтаксис, семантика, перевод. [Електронний ресурс] // Режим доступу : http://public.uic.rsu.ru/~skritski/scourses/Transl/index.html 2013.

118. Кузнецов С. Переносимость и интероперабельность информационных систем и международные стандарты [Текст] / С. Кузнецов // Computer World. 1996. № 4. — С. 5-6.

119. Леоненков А.В. Нечёткое моделирование в среде MatLab и fuzzyTech [Текст] / А. В. Леоненков. — СПб : БХВ-Петербург, 2003. — 736 с.

120. Лоскутов А.Ю. Основы теории сложных систем [Текст] / А. Ю. Лоскутов, А. С. Михайлов —М.-Ижевск : НИЦ «Регулярная и стохастическая динамика», 2007. — 620 с.

121. Майоров С.А. Основы теории вычислительных систем [Текст] / С.А. Майоров, Г.И. Новиков, Алив Т.И. и др. Учеб. пособие для ВУЗов. — М : Высшая школа, 1978. — 408 с.

122. Меньшиков Н.Я. Надёжность железнодорожных систем автоматики [Текст] / Н.Я. Меньшиков, А.И. Королёв, Р.Ш. Ягудин —М: Транспорт, 1976. — 215 с.

123. Месарович М. Теория иерархических многоуровневых систем : Пер. с англ. [Текст] / М. Месарович, Д. Мако, И. Такахара. — М. : "Мир", 1973. — 344 с.

124. Мішечкін В. Історія автоматизації перевізних процесів / В. Мішечкін // 2013. - [Электронный ресурс]. - Режим доступа : URL: http://www.youtube.com/watch?v=FzBG7-Pkx4M&feature=relmfu/. 2013.

125. Нейгел К. C# 2005 и платформа .NET 3.0 для профессионалов [Текст] / Кристиан Нейгел, Билл Ивьен, Джей Глинн, Морган Скиннер, Карли Уотсон. — М : Диалектика, 2008. — 1376 с.

126. Нечипоренко В.И. Структурный анализ систем (эффективность и надёжность) [Текст] / В. И. Нечипоренко. — М. : Сов. радио, 1977. — 216 с.

127. Плотникова А. АСК ВП УЗ-Е – 20 дней спустя / А. Плотникова // "Магистраль", 01.08.2012. - [Электронный ресурс]. - Режим доступа : URL: www.magistral-uz.com.ua/. 2012.

128. Половко А.М. Основы теории надёжности [Текс] / А. М. Половко, С.

В. Гуров. — Спб. : БХВ-Петербург, 2008. — 704 с.

129. Растригин Л.А. Адаптация сложных систем [Текст] / Л.А. Растригин. — Рига : Зинатне, 1981.

130. Савицкий А.Г. Комплексная система автоматизированного управления сортировочной станцией [Текст] / А. Г. Савицкий // Евразия Вести. 2004. № XI.

131. Савицкий А.Г. Комплексная система автоматизированного управления сортировочным процессом [Текст] : автореф. дис. .... к.т.н. 05.22.08 / Савицкий Александр Григорьевич. — Москва : Московский государственный университет путей сообщения (МИИТ), 2005. — 24 с.

132. Самков А.Н. К вопросу о выборе УВК для АСУРСГ [Текст] / А.Н. Самков, Е.М. Шафит // Труды ДИИТа. 1980. № 211/9. — С. 11-17.

133. Сафрис Л.В. Концептуальная модель автоматизированной системы управления роспуском составов на горке [Текст] / Л.В. Сафрис // Автоматизированные системы управления технологическими процессами на железнодорожных станциях. Межвуз. сб. научн. трудов. - Днепропетровск: изд. ДИИТа. 1981. № 218/10. — С. 8-14.

134. Сафрис Л.В. Об одном критерии автоматического выбора оптимальной скорости роспуска [Текст] / Л.В. Сафрис // Труды ДИИТа Т. 121/3. — Днепропетровск : ДИИТ, 1971.

135. Сафрис Л.В. Стратегия целевого торможения при автоматизации сортировочной горки [Текст] / Л.В. Сафрис // Труды ДИИТа Т. 162/6. — Днепропетровск : ДИИТ, 1974.

136. Сергиенко Н.И. Проектный подход при внедрении информационных технологий и систем для железнодорожного транспорта Украины [Текст] / Н.И. Сергиенко, О.Л. Зиненко // Современные информационные технологии на транспорте, в промышленности и образовании. Материалы Междунар. научно-практ. конф., г. Днепропетровск, 18-19 апреля 2013 г. 2013. — С. 87.

137. Сидорова Е.Н. Автоматизированные системы управления в эксплуатационной работе [Текст] / Е. Н. Сидорова. — М. : Маршрут, 2005. — 560 с.

138. Стукалов Е.А. Определение параметров потока инициативных сигналов, поступающих в УВК АСУ расформированием составов на сортировочной горке [Текст] / Е.А. Стукалов // Труды МИИТа. 1983. № 719. — С. 93-99.

139. Трифонов П.В. Информатика. Построение и анализ алгоритмов :

Учебное пособие для ВУЗов [Текст] / П.В. Трифонов. — Санкт-Петербург : Питер, 2007. — 95 с.

140. Трутнев Д.Р. Архитектуры информационных систем. Основы проектирования : Учебное пособие [Текст] / Д. Р. Трутнев. — СПб : НИУ ИТМО, 2012. — 66 с.

141. Туманова Т. Единый Центр обработки данных заработал в Укрзализныце [Текст] / Т. Туманова // УНН. - 2012. - [Электронный ресурс]. - Режим доступа : URL: http://www.unn.com.ua/. 2012.

142. Федюшин Ю.М. Информатизация железнодорожного транспорта Украины / Ю. М. Федюшин // Информационно-управляющие системы на железнодо-рожном транспорте. 1997. № 4. — С. 3-5.

143. Фонарев Н.М. Автоматизация процесса расформирования составов на сортировочных горках [Текст] / Н.М. Фонарев. — М : Транспорт, 1971. — 271 с.

144. Фритч В. Применение микропроцессоров в системах управления : Пер. с нем. [Текст] / В. Фритч. — М. : Мир, 1984. — 464 с.

145. Хакен Г. Синергетика. Иерархии неустойчивостей в самоорганизующихся системах и устройствах. [Текст] / Г. Хакен

—М. : Мир, 1985.

146. Хетагуров Я.А. Проектирование информационно-вычислительных комплексов [Текст] // Я.А. Хетагуров, Ю.Г. Древс —М. : Высш. шк., 1987. — 280 с.

147. Цейтлин С.Ю. Типовые проектные решения для создания АСУ ВП УЗ-Е [Текст] / С. Ю. Цейтлин, В. К. Башлаев // Тезисы Международной научно-практической конференции "Современные информационные технологии на транспорте, в промышленности и образовании" (15.05.2008-16.05.2008), Днепропетровск. 2008. — С. 31-32.

148. Цилькер Б.Я. Организация ЭВМ и систем : Учебник для ВУЗов [Текст] / Б.Я. Цилькер, С.А. Орлов. — Санкт-Петербург : Питер, 2006. — 654 с.

149. Чистяков Л.С. Модифицированная HIPO-технология разработки больших программных комплексов [Текст] / Л.С. Чистяков // Прикладная информатика. 2006. № 6. — С. 64.

150. Шастова Г.А., Коёкин А. И. Выбор и оптимизация структуры информационных систем [Текст] / Г. А. Шастова, А. И. Коёкин —М. : Энергия, 1972. — 256 с.

151. Шафит Е.М. Исследование влияния надежности на показатели

качества функционирования сложных систем [Текс] / Е.М. Шафит, А.Н. Самков, В.Н. Рыбцов // Автоматизированные системы управления технологическими процессами на железнодорожных станциях: Межвуз. сб. научн. тр. - Днепропетровск : ДИИТ. 1984. — С. 3-10.

152. Шафит Е.М. Экономическая эффективность и надежность систем автоматического управления сортировочным процессом с управляющей ЦВМ [Текст] / Е.М. Шафит , А.Н. Самков // Труды ДИИТа. 1971. № 129/3. — С. 3-20.

153. Шлеер С. Объектно-ориентировочный анализ: моделирование мира в состояниях : Пер. с англ. [Текст] / С. Шлеер, С. Меллор. — Киев : Диалектика, 1993. — 240 с.

154. Юнг М. Сименс – партнёр для автоматизации сортировочных станций [Текст] / М. Юнг, О. В. Подсосонная // Автоматика, связь, информатика. 2009. № 1. — С. 51-52.

155. Baun C. Cloud Computing. Web-Based Dynamic IT Services / C. Baun. — Berlin : Springer-Verlag, 2011. — 108 p.

156. Cheng C.H., and Mon, D. L. . Fuzzy system reliability analysis by interval of confidence // Fuzzy Sets and Systems. 1993. Т. 56. № 1. — С. 29-35.

157. Dave P. System Design Strategies // An ESRI White Paper. February. 2003.

158. Automation Device and Computing 2012-2013 [Електронний ресурс] / Режим доступу : http://support.advantech.com.tw/ OnlineResources/ SubIndex.aspx?bu=99B2E2BE-A7E3-4C58-9E35-B4F48A952AC7&type =e Catalog. : Advantech, 2013.

159. Definition issue des travaux du groupe de travail Interop de I'AFUL [Электронный ресурс] // Режим доступа: http:// http://aful.org/gdt/interop. 2013.

160. The MSR 32 microcomputer system. Greater efficiency and safety for cargo transport [Электронный ресурс] / Siemens AG // Режим доступа : http://www.siemens.com/mobility. 2008.

161. Produre, method and organization for the development of international Computer system within the UIC. PROMETEO / UIC Code 902. — Paris : Int. Union of Railways, 1993. — 138 p.

162. The Protégé Ontology Editor [Электрон. ресурс] / Режим доступа: http://protege.stanford.edu/. — Manchester, 2013.

163. Real Time Systems Modeling, Design and Applications. AMAST Series in Computing // World scientific. 2007. Т. 8. — С. 37-62.

164. A Survey of System Development Process Models CTG.MFA – 003

Models for Action Project : Developing Practical Approaches to Electronic Records Management and Preservation. — Center for Technology in Government University at Albany : SUNY, 1998. — 13 p.

165. SYSTEM DEVELOPMENT METHODOLOGY (SDM) September 6, Version 2, 02.18.2005. — NH Office of Information Technology : Agency Software Division (ASD) Effective, 2006. — 35 p.

166. Adamek J., Trnkova V. Automata and algebras in categories. : Kluwer Academic Publisher. 1989.

167. Arbib M.A. Arrows, structures and functors : The categorical imperative / M. A. Arbib & E. G. Manes. : Academic Press, 1975. — 320 p.

168. Arbib M.A. Theories of Abstract Automata // Series in Automatic Computation — Englewood Cliffs, NJ : Prentice Hall, 1969.

169. Atanassov K. Intuitionistic fuzzy sets / K. Atanassov // Fuzzy Sets and Systems. 1986. № 20. — С. 87 - 96.

170. Baeten J.C.M. Real time process algebra / J. C. M. Baeten & J. A. Bergstra // Formal Aspects of Computing. 1993. Т. 3. — С. 142-188.

171. Bertoli M. The JMT simulator for performance evaluationof non-product-form queueing networks / M. Bertoli, G. Casale, G. Serazzi // Proc. of the 40th Annual Simulationymposium (ANSS). 2007. — С. 3-10.

172. Brandin B.A. Supervisory control of timed discrete-event systems / B. A. Brandin & M. W. Wonham // IEEE Transactions on Automatic Control. 1994. Т. 39(2). — С. 329-343.

173. Buchi R.J. Finite automata, their algebras and grammars / R. J. Buchi. : Springer-Verlag, 1989. — 434 p.

174. Cai K.Y. Posbist reliability behavior of typical systems with two types of failure / K.Y. Cai, C.Y. Wen, M. L. Zhang // Fuzzy Sets and Systems. 1991. № 43. — С. 17 - 32.

175. Chang J.R., Chang K. H., Liao, S. H., and Cheng, C. H. . The reliability of general vague fault-tree analysis of weapon systems fault diagnosis // Soft Computing. 2006. Т. 10. — С. 531-542.

176. Chapman W.L. System design is an NP-complete problem / W. L. Chapman, J. Rozenblit, and A. T. Bahill // Systems Engineering, The Journal of INCOSE. 2001. № 4(3). — С. 222-229.

177. Chen S. An overview of quality of service routing for next-generation high-speed networks: problems and solutions / S. Chen, K. Nahrstedt // IEEE Networks 1998. № 12. — С. 64-79 p.

178. Chen S.M. Analyzing fuzzy system reliability using vague set theory / S. M. Chen // International Journal of Applied Science and Engineering. 2003. № 1. — C. 82 - 88.

179. Chen S.M. Arithmetic operations between vague sets / S.M. Chen // Proceedings of the International Joint Conference of CFSA/IFIS/SOFT'95 on Fuzzy Theory and Applications, Taipei, Taiwan, Re¬public of China. 1995. — C. 206 - 211.

180. Curtain R. Functional analysis in modern applied mathematics / R. Curtain, A. J. Pritchard : Academic Press, 1997.

181. Ein-Dor P. Grosh's law re-revisited: CPU power and the cost of computation // Communication of the ACM. 1985. T. 28. № 2. — C. 142-151.

182. Franke E. Zur Aufwadsbewertung dezentraler bzw. zentraler Prozesrechnerstrukturen / Forschungsinstitut // AEG-TELEFUNKEN Ulm. 1986. — C. 146-159.

183. Gau W.L. Vague sets / W. L. Gau, D. J. Buehrer // IEEE Transactions on Systems, Man, and Cybernetics. 1993. № 23. — C. 610 - 614.

184. Gawlick R. Liveness in timed and untimed systems / R. Gawlick, R. Segala, J. F. Sogaard-Anderson, N. Lynch & B. Lampson // Technical report TR-587, MIT Computer Science Laboratory. 1993. — C. 289.

185. Gottlieb J. Prüfer numbers: A poor representation of spanning trees for evolutionary search / Jens Gottlieb, Bryant A. Julstrom, Günther R. Raidl, and Franz Rothlauf // Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2001). 2001. — C. 343–350.

186. Gruber T.R. The role of common ontology in achieving sharable, reusable knowledge bases [Текст] / J.A. Allen, R. Fikes, T.R. Gruber, E. Sandewell – eds. Morgan Kaufmann // 1991. — C. 601-602.

187. Gumzej R. Real-time Systems' Quality of Service. Introducing Quality of Service Considerations in the Life-cycle of Real-time Systems / R. Gumzej, W. A. Halang. — London : Springer-Verlag London Limited, 2010. — 145 p.

188. Ionescu D. Designing supervisor for real-time systems. Theories and experiences for real-time system development / D. Ionescu. : World Scientific Publishing Company, 1994. — P. 103-128.

189. J.-B. Kruger. Bewertungsmasstabe fur den Zuverlassigkeitsvergleich von zentral und dezentral arbeitenden Prozesautomatisierungssystemen // Regelungstechnishe Praxis. 1978. T. 20. № 8. — C. 225-252.

190. K. E. Knight. Changes in computer perfoirmance. A historical view // Datamation. 1966. № 9. — C. 40-54.

191. Kalman R. Mathematical theory of dynamic systems / P. Falb & M. A. Arbib. — New York : Academic Press, 1969.

192. Kleinrock L. Distributed systems. Special issue // Communication of the ACM. 1985. Т. 28. № 11. — С. 1200-1213.

193. Kosolapov A. Application of Network Information Technologies to Integration of Educational Process Control and Manage-ment in Technical University / O. Pshinko, B. Bodnar, A. Raspopov, A. Kosolapov // International Conference on Engineering Education, ICEE'99, Ostrava - Prague: VSB - Technical University of Ostrava. 1999. — С. 12.

194. Krogstie J. A. L. Opdahl, S. Brinkkemper Conceptual Modelling in Information Systems Engineering / J. Krogstie, A. L. Opdahl, S. Brinkkemper. — Berlin : Springer-Verlag Berlin Heidelberg, 2007. — 356 p.

195. Kumar A., Yadav, S. P., Kumar, S. Fuzzy System Reliability Using Different Types of Vague Sets // Int. Journal of Applied Science and Engineering. 2008. Т. 6. № 1. — С. 71-83.

196. Landau A. A METHODOLOGICAL FRAMEWORK FOR BUSINESS-ORIENTED MODELING OF IT INFRASTRUCTURE / A. Landau, S. Wasserkrug, D. Gilat, IBM Haifa Research Lab University Campus Carmel Mountains Haifa, 31905, ISRAEL // Proceedings of the 2004 Winter Simulation Conference R .G. Ingalls, M. D. Rossetti, J. S. Smith, and B. A. Peters, eds. . 2004. — С. 1-9.

197. Leeman H., Dedene G. Grosch's law: a statistical illusion? / Onderzoeksrapport NR 9631 // Katholieke Universiteit Leuven. 1996. — С. 1-16.

198. Lin J.Y. Temporal Logic Approach to the Analysis and Synthesis of Discrete Event Systems. PhD thesis / J. Y. Lin. — University of Ottawa, 1993.

199. Liu T.S. The Role of a Maintenance Processor for a General-Purpose Computer System / T.S. Liu // IEEE Trans Comput. 1984. Т. 33. — С. 507-517.

200. Lixia H. A Nove Genetic Algorithm for the Degree-constrained Minimum Spanning Tree Problem / Lixia Han // Computer Engineering and Applications. 2006. Т. 42(31). — С. 37-42

201. Manna Z. The temporal logic of reactive and concurrent systems / Z. Manna & A. Pnueli. : Springer-Verlag, 1992. — 350 p.

202. Ostroff J.S. Temporal logic for real-time systems / J. S. Ostroff — NewYork : John Wiley and Sons, 1989.

203. Panzieri F. Системы реального времени: основные понятия [Електронний ресурс] / F. Panzieri, R. Davoli. Laboratory for Computer Science. University of Bologna. Режим доступу : ftp:// ftp.cs.unibo.it:/pub/TR/UBLCS //

Техническое описание UBLCS-93-22. Октябрь. 1993.

204. René J. Chevance Server Architectures. Multiprocessors, Clusters, Parallel Systems, Web Servers, and Storage Solutions / J. René. — USA : Elsevier Digital Press, 2005. — 709 p.

205. Segala R. Modeling and Verification of Randomized Distributed Systems / R. Segala // PhD thesis, Massachusetts Institute of Technology. 1995. — C. 1-123.

206. Shan T.C. Towards a Systematic Method for Solutions Architecting. Handbook of research on modern systems analysis and design technologies and applications / Tony C. Shan, Winnie W. Hua. — N.Y. : IGI Global, 2009. — P. 1-22.

207. Singer D. A fuzzy set approach to fault tree and reliability analysis // Fuzzy Sets and Systems. 1990. T. 34. № 2. — C. 145-155.

208. Sogaard-Anderson J.F. Correctness of communication protocols / J. F. Sogaard-Anderson, N. Lynch & B. Lampson // Technical report TR-589, MIT Computer Science Laboratory. 1993. — C. 4-17.

209. Stephen Blacketer S. ISO 9001:2001 Quality Management System - System Development Methodologies MBP5002. 22nd November 2001, вер. 1.0.2. / Stephen Blacketer —London : ISO, 2001. — 6p.

210. Streeter D.N. Centralization or dispersion of computing facilitien // IBM System Journal. 1973. № 3. — C. 283-301.

211. Topfer Y., Reinig G. Automatisierungsstrukturen, ihre Systematishe und Rechnergestutztg Gestaltung / Technische Universitat Dresden // Wissenschaftliche Berichte. 1986. T. 35. № 1. — C. 130-138.

212. W. Fritzsch. Anlagenstrukturen und Steuerunskonzepte mit Mikroprozesrechnern // Messen-Steuern-Regeln. 1977. T. 20. № 12. — C. 662-667.

213. Yzhak R. Distributed computing system as an altrnative to a central computing center. Cost effective analysis // 14 Conf. Elec. and Electron. Eng., Israel, Tel-Aviv, 26-28 March. 1986. — C. 1.2.4/1-1.2.4/6.

214. Zadeh L.A. Fuzzy sets // Inform. Control. 1965. T. 8. № 3. — C. 338-353.

215. Zarecky S. The newest trends in marshalling yards automation / S. Zarecky, J. Grun, J. Zilka // Transport problem. 2008. T. 3. № 4, Part 1. — C. 87-95.

216. Zhang R. Fuzzy Control of Queuing Systems / R. Zhang, Y.A. Phillis, V.S. Kouikoglou. — London : Springer-Verlag, 2005. — 175 p.

В данной научной монографии впервые предложен комплекс моделей, методов и методик решения задач концептуального проектирования компьютерных систем, работающих в реальном масштабе времени CoDeCS. Особое внимание уделено ресурсосберегающим методам формирования рациональных технических структур распределенных информационно-управляющих систем (РИУС). Применение предложенных математических средств иллюстрируется примерами системного проектирования реальных РИУС автоматизации процессов на железнодорожных сортировочных станциях. Многие технические решения и программы автоматизации расчетов характеристик РИУС на ранних стадиях их проектирования защищены авторскими свидетельствами. Монография будет полезна как молодым исследователям (студентам старших курсов, магистрам и аспирантам), так и специалистам, разрабатывающим сложные РИУС в различных отраслях промышленности и на транспорте.

Косолапов Анатолий Аркадьевич (Anatolii A. Kosolapov).Профессор, доктор технических наук, профессор кафедры ЭВМ Национального университета железнодорожного транспорта (г. Днепр, Украина). Область научных интересов: проектирование автоматизированных систем контроля и управления, системы искусственного интеллекта, WEB-технологии, философия АйТи.

978-620-0-08208-4