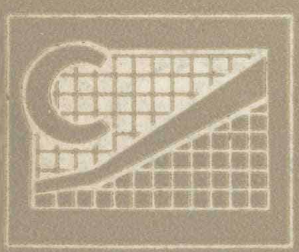
Алгоритмы и организация решения экономических задач



АЛГОРИТМЫ И ОРГАНИЗАЦИЯ РЕШЕНИЯ ЭКОНОМИЧЕСКИХ ЗАДАЧ

СБОРНИК СТАТЕЙ
ПОД РЕДАКЦИЕЙ
В. М. САВИНКОВА

Выпуск
14

МОСКВА
«СТАТИСТИКА»
1980

ББК 32.973
А45

А $\frac{30502-036}{008(01)-80}$ 77—80 1502000000

ПРЕДИСЛОВИЕ

С 1973 г. сборник «Алгоритмы и организация решения экономических задач» оперативно знакомит читателей с новыми теоретическими и практическими результатами, развивающими одно из важнейших научных направлений современности — применение вычислительной техники в обработке информации. Четыре традиционных раздела сборника — «Методология и организация решения экономических задач», «Проектирование и моделирование систем обработки данных», «Языки, системы программирования, операционные системы», «Техническое обеспечение систем обработки данных» — за истекшие 8 лет дали путевку в жизнь многим пакетам прикладных программ, языкам и системам программирования, методам и технологии проектирования информационных систем.

На страницах сборника печатаются статьи ведущих советских ученых-программистов, конструкторов ЭВМ и разработчиков операционных и информационных систем, организаторов отечественной индустрии обработки информации.

Своевременная публикация работ по программному обеспечению банков данных во многом способствовала внедрению в практику проектирования автоматизированных систем новой технологии, основанной на применении систем управления базами данных («Банк», «Ока», «Набоб»).

Фактическая проблематика статей сборника подпадает под современное определение *информатики* как самостоятельной научной дисциплины. Прикладной характер опубликованных материалов позволяет отнести их к прикладным разделам информатики, или к *прикладной информатике.*

На этом основании издательство «Статистика» приняло предложение редакционной коллегии о переименовании сборника. Последующие выпуски сборника будут называться «Прикладная информатика» *. Такое название, по нашему мнению, точнее определяет проблематику сборника. И мы надеемся, что к моменту выхода сборника с новым названием термин «информатика» завоюет все-

* В выборе нового названия сборника принимали участие, кроме членов редколлегии, постоянные авторы сборника, представители Рабочей группы по программному обеспечению банков данных и др.

общее признание и заменит бытующее в настоящее время «вычислительное дело».

Также изменяются и названия разделов сборника: «Системы информатики», «Программные средства информатики», «Технические средства информатики», «Анализ применений».

Преемственность указанных выше новых разделов традиционной тематике сборника лучше всего подтверждается содержанием настоящего выпуска.

В статье Г. К. Григаса «Кластероподобный стиль программирования на языке ПЛ/1» описывается, как, оставаясь полностью в рамках языка ПЛ/1, можно программировать в стиле языка с абстрактными типами данных.

Ценность статьи Р. А. Маркевичюса «Реализационные свойства абстрактных типов данных» состоит в том, что в ней показывается, как со сравнительно небольшими трудовыми затратами можно создать расширяемую систему программирования с абстрактными типами данных.

Указанные статьи Г. К. Григаса и Р. А. Маркевичюса рекомендованы к публикации программным комитетом Всесоюзного семинара «Программное обеспечение банков данных» и их можно рассматривать как заметное продвижение к практическому программированию с абстрактными типами данных.

Зарождение структурного программирования (точнее, структурного подхода) связывают обычно с публикацией в марте 1968 г. письма Дейкстры «О вреде оператора GO TO» *.

Основная посылка структурного программирования — практически осуществимое доказательство правильности программы, которое возможно на уровне ее структуры, описанной псевдокодом или другими выразительными средствами. Несмотря на такие очевидные достоинства, структурное программирование еще неполно представлено в массовых изданиях.

В статье Ф. Я. Дзержинского «Язык для проектирования структуризованных программ» описываются правила использования псевдокода — частично формализованной нотации, предназначенной для тех же целей, что и язык блок-схем (наглядного представления схем алгоритмов и программ), но соответствующей принципам и стилю структурного подхода в программировании и имеющей текстовый характер.

В статье В. В. Митрофанова и др. «Автоматизация технологии редакционно-издательского процесса на базе ЕС ЭВМ и фотонаборной техники» описывается пакет прикладных программ «Астра» и разработанная с учетом его применения технология оперативного выпуска и перевыпуска текстовой документации.

Система «Астра» позволяет с высокой производительностью вносить изменения в текст, не требуя при этом дополнительных операций по перекомпоновке. При этом обеспечивается высокое качество исполнения документов.

* Dijkstra E. W. GO TO statements considered harmful. Comm. of ACM, v. 11, 1968, № 3, p. 147—148.

Впервые в отечественной практике применения ЕС ЭВМ предложена диалоговая система коллективного пользования, обеспечивающая технологический цикл обработки текстовой информации вплоть до получения перфоленты для управления фотонаборным автоматом.

В предыдущих выпусках сборника читатели познакомились с рядом статей по проблеме СУБД. В настоящем выпуске эта, ставшая традиционной, тема развивается в статье В. И. Будзко. Автор предлагает математический аппарат выбора стратегии поддержания целостности базы данных, основываясь на анализе практически всех отечественных и наиболее массовых зарубежных СУБД.

В связи с комплектованием моделей ЕС ЭВМ дисками в 29 и 100 Мбайт становится возможным создание баз данных очень больших размеров. В условиях многолетней эксплуатации такие базы подвержены воздействию неисправной аппаратуры, вызывающей искажения в пересылаемых сообщениях. Чтобы не допустить в таких условиях нарушения целостности, СУБД должны располагать совершенным механизмом, обеспечивающим повторение процесса актуализации базы с некоторой точки до ее правильного завершения. Результаты, приводимые в статье В. И. Будзко, существенно расширяют арсенал средств администратора базы данных и будут способствовать нормальному функционированию автоматизированных информационных систем с СУБД типа «Ока» и др.

Из остальных статей отметим еще только статью Ф. В. Широкова, в которой в популярной форме излагаются основные особенности программирования на языке APL. По ряду причин (главным образом из-за отсутствия дисплеев и устройств подготовки данных с соответствующей клавиатурой) язык APL еще недостаточно используется на машинах Единой системы и малых машинах. В то же время ряд свойств этого языка сохраняют за ним (как за языком Паскаль и др.) хорошие перспективы.

Надеемся, что наш сборник под новым названием «Прикладная информатика» полнее удовлетворит интересы различных категорий читателей.

1 ПРОЕКТИРОВАНИЕ И МОДЕЛИРОВАНИЕ СИСТЕМ ОБРАБОТКИ ДАННЫХ

В. И. БУДЗКО

ОБЕСПЕЧЕНИЕ ЦЕЛОСТНОСТИ АИС

Обеспечение целостности автоматизированной информационной системы (АИС) является одной из главных задач, которую приходится решать при разработке системы и в процессе ее эксплуатации [2, 16, 19]. Эта проблема значительно усложняется при наличии больших объемов данных, составляющих сотни миллионов или миллиарды знаков. Простое копирование базы данных (БД) такого объема потребует многих часов работы ЭВМ.

Другой важный момент состоит в том, что БД современной АИС имеет сложную структуру, содержит взаимосвязанные элементы данных разного типа и рассчитана на эксплуатацию в течение длительного времени. В таких условиях огромные ресурсы, затрачиваемые на ввод и обновление данных, могут оказаться невосполнимыми в случае разрушения БД, а само разрушение может оказаться катастрофическим для всей АИС, если не будут предусмотрены специальные средства восстановления [9].

Многие системы так и не были внедрены после завершения разработки из-за отсутствия необходимых средств восстановления целостности [22]. Под целостностью АИС понимается ее способность в данный момент выполнять все действия в соответствии с некоторыми вполне определенными правилами. Целостность АИС включает целостность базы данных, т. е. обеспечение возможности восстановления БД в некоторое пригодное для работы состояние, и функциональную целостность, которая связана со способностью системы после возникновения отказа продолжать нормальное функционирование (например, восстановление очередей сообщений, вывод не выданных из-за отказа сообщений, продолжение прерванных диалогов и т. д.).

Целостность БД можно условно разбить на синтаксическую и семантическую. Синтаксическая целостность предполагает сохранение данных, введенных в БД, и отношений между ними; семантическая связана с правильностью значений самих данных и обеспечивается программами логического контроля информации при ее вводе в систему (например, контроль по диапазону, алфавиту и т. д.). Таким образом, можно выделить три уровня целостности АИС: синтаксический, семантический и функциональный.

Целостность АИС может быть нарушена в результате отказа, т. е. ситуации, при которой АИС не может продолжать нормально функционировать без выполнения некоторых процедур восстановления. Поэтому для обеспечения восстановления целостности АИС после возникновения отказа должны существовать специальные средства восстановления, которые органически связаны и сбалансированы с аппаратом управления данными [4]. Имеется множество типов отказов и соответственно типов процедур восстановления.

Для выполнения различных типов процедур восстановления средства восстановления выполняют определенные действия по поддержанию различных типов избыточных данных. Чем представительнее множество отказов, которое «покрывается» средством восстановления, тем сложнее это средство. Например, в одной из наиболее «надежных» систем управления базами данных (СУБД) IMS [7] программы восстановления составляют 17% общего объема программ [21], который оценивается в 500 тысяч команд на языке Ассемблер.

Во всех основных коммерческих СУБД, таких как IMS, IDMS, ADABAS, SYSTEM 2000 [6, 7, 8, 21], существуют средства восстановления высокого уровня, которые предусматриваются и в специализированных СУБД (ACP [18] или CMIC [11]) и во вновь создаваемых, например в системе R [5, 14]. В отечественных СУБД, таких, как «Ока», «Банк», СИОД, содержатся развитые средства обеспечения синтаксической и функциональной целостности, например в СУБД «Ока», и синтаксической целостности — в СУБД «Банк», СИОД.

В настоящей статье основное внимание уделяется синтаксической целостности БД и функциональной целостности системы. Вопросы семантической целостности не рассматриваются, так как, по мнению автора, они находятся в основном вне сферы влияния СУБД и должны решаться на уровне прикладных программ.

Ядром современной АИС является СУБД, причем в зависимости от требований, предъявляемых к АИС, класса решаемых задач и условий использования возможны следующие ситуации:

может потребоваться разработка собственной уникальной СУБД;

вполне приемлемым окажется применение уже существующей СУБД, но с реализацией информационных функций системы специально разработанным комплексом прикладных программ;

СУБД может покрыть все требования, которые предъявляются к АИС.

Но во всех случаях предстоит решать вопросы эксплуатации системы.

Таким образом, можно выделить следующие основные задачи, связанные с обеспечением целостности АИС:

анализ возможных типов отказов и необходимых типов процедур восстановления;

разработку средств обеспечения целостности АИС;

оптимизацию работы средств обеспечения целостности;

конструирование АИС с учетом обеспечения целостности;
разработку технологии эксплуатации системы с учетом обеспечения целостности.

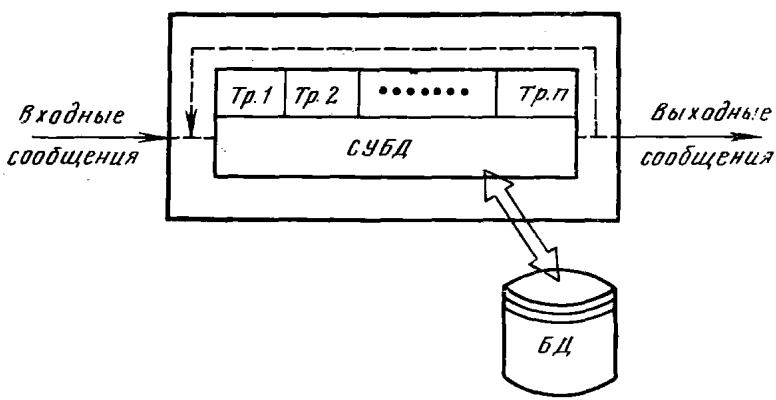
Выбор конкретной организации средств обеспечения целостности, а также типов процедур восстановления среди уже существующих зависит от применяемого критерия эффективности. Среди широко используемых критериев можно выделить следующие:
минимум средней стоимости АИС в единицу времени [13, 15];
максимум суммарной пропускной способности системы [13, 23];
минимум среднего времени отклика системы [3, 10].

По выбранному критерию должна оптимизироваться работа средств восстановления. Разные способы организации средств восстановления или различные типы процедур восстановления внутри одного средства следует сравнивать в условиях оптимальной по выбранному критерию работы средств восстановления.

Модель целостности АИС

Ниже рассматривается класс АИС, которые можно описать следующей упрощенной моделью: «черный ящик», вырабатывающий на определенные входы необходимые выходы. Для выполнения таких действий внутри «черного ящика» имеются необходимые программы, которые через СУБД могут обращаться к данным, расположенным в БД. Входы системы разбиваются на четко определенные группы — входные сообщения и заставляют АИС выполнять определенные действия. При этом для получения необходимого выхода используется общая БД. Выход системы называется выходным сообщением, которое может выдаваться пользователю или поступать на вход и становиться входным сообщением.

В системе допускается одновременная обработка нескольких сообщений, т. е. мультипрограммирование и мультидоступ к БД, включая одновременное обновление. Действия системы по обработке одного входного сообщения называются транзакцией. Таким образом, в системе могут одновременно выполняться несколько транзакций. Различаются два типа транзакций: изменяющие БД и неизменяющие БД (см. рисунок).



База данных состоит из массивов и связей между ними; массив содержит однотипные записи иерархической структуры, записи из групп, группы из полей. Частным случаем этой структуры является реляционная БД, когда запись состоит из одной группы, а группа содержит неповторяющиеся поля. Транзакция может одновременно взаимодействовать как с одним, так и с несколькими массивами БД.

Выделяются три состояния целостности БД, в которых она может находиться после возникновения отказа и отработки средств восстановления [21]:

точное, когда БД содержит все данные, помещенные в нее пользователем, и не содержит данных, удаленных из нее пользователем;

действительное, если БД содержит данные, являющиеся частью точного состояния, т. е. БД не содержит неправильных связей или данных, но часть информации может быть потеряна (например, все изменения, которые были внесены несколькими транзакциями);

непротиворечивое, если БД в действительном состоянии и содержащиеся в ней данные удовлетворяют пользовательским ограничениям непротиворечивости. Можно полагать, что точное состояние является непротиворечивым состоянием.

В реальных условиях непротиворечивость БД постоянно нарушается при выполнении транзакций первого типа. Пусть массив M в момент t находится в непротиворечивом состоянии $D(t)$ и за время Δt в M осуществляет изменения транзакция первого типа.

В момент $t+\Delta t$ M перейдет в новое непротиворечивое состояние $D(t+\Delta t)$. Если за время Δt возникнет отказ, то непротиворечивость M будет нарушена.

Транзакции первого типа являются единицами целостности, они переводят БД из одного непротиворечивого состояния в другое. Прерывание такой транзакции равносильно нарушению непротиворечивости БД. А работа других транзакций с массивом, который обновляет данная транзакция, означает работу с массивом, непротиворечивость которого нарушена.

Средства обеспечения целостности БД должны предусматривать в случае отказа возможность возврата массива M в непротиворечивое состояние $D(t)$ и предоставлять аппарат, использование которого будет гарантировать, что каждая транзакция работает с непротиворечивыми данными.

Отказ АИС возникает, когда система не удовлетворяет своим спецификациям, т. е. не может нормально продолжать свою работу. Восстановление АИС в качестве основного элемента включает восстановление непротиворечивости БД.

Вторым элементом восстановления АИС является восстановление всех прерванных транзакций, а также очередей заявок на новые транзакции (очередей сообщений) и очередей выходных сообщений, что в конечном итоге должно привести БД в точное состояние.

Непротиворечивость БД при отсутствии отказов

Непротиворечивость БД нарушается во время выполнения транзакции, а поскольку допускается одновременное выполнение нескольких транзакций с доступом к одной БД, необходимо обеспечить в данный момент непротиворечивость той части БД, к которой обращается транзакция.

Например, пусть одновременно выполняются две транзакции Возможна ситуация, когда сразу после получения от СУБД одной из транзакций некоторой записи эта же запись поступает к другой транзакции. Если обе транзакции осуществляют изменение записи, например увеличение значения некоторого поля, то в БД занесется изменение, произведенное только одной из них. Подобная ошибка может не привести к немедленному отказу и оставаться необнаруженной в течение длительного времени.

Для предотвращения таких ошибок в современных СУБД предусматриваются средства блокировки. Если СУБД предоставляет средства, при которых блокировка контролируется пользователем, то потенциально эта блокировка короче системной, так как пользователь (транзакция) знает семантику данных. С другой стороны, такой тип блокировки требует сложных и потенциально ненадежных прикладных программ. В то же время те СУБД, которые осуществляют автоматическую блокировку и обеспечивают защиту от рассмотренного в примере типа нарушений непротиворечивости, все же возлагают на пользователя защиту от других источников нарушения непротиворечивости, например могут автоматически блокироваться обновляемые записи, но не блокироваться читаемые. Существуют теоретические и практические результаты по вопросу блокировки доступа к элементам данных.

Рассмотрим основные механизмы блокировки доступа к БД, которые применяются в современных СУБД.

В средствах автоматической блокировки учитывается требуемый транзакцией режим работы с элементами данных БД, определяемый либо в момент открытия БД командой OPEN, либо при планировании транзакции. Режим работы определяет допустимые намерения других транзакций, с которыми данная транзакция может одновременно функционировать. Наиболее общие режимы и их связь с намерениями приведены в табл. 1

Таблица 1

Намерение	Режим работы		
	монопольный	защищенный	разделяемый
Только поиск	Нет	Да	Да
Поиск и обновление	Нет	Нет	Да

Намерения транзакции могут выражаться по отношению к различным элементам структуры данных: массиву, типу группы и типу поля, а также по отношению к конкретным экземплярам элементов данных. В общем случае чем ниже уровень задания намерений, тем меньше конфликтов намерений, т. е. тем большее количество транзакций могут одновременно выполнять свои намерения. В то же время на глубину допустимых намерений существенное влияние может оказать специфика конкретной системы.

Автоматическая блокировка в существующих СУБД в основном выполняется на все время работы транзакции. Такой способ блокировки называется длинной блокировкой Сама транзакция может и не «знать» о действиях СУБД, так как в большинстве реализаций намерения определяются во внешних по отношению к прикладным программам описаниях администратором базы данных. Длинная блокировка реализуется в системах «Ока», «Набоб», IDMS, IMS.

При неавтоматической блокировке СУБД осуществляет блокировку элемента данных только после выдачи транзакцией специальной команды. Блокировка продолжается до получения СУБД сигнала разблокировки, которым может служить:

завершение выполнения команды обновления;
выдача транзакцией команды разблокировки;
чтение транзакцией другой записи;
завершение транзакции (нормальное или аварийное);
запись контрольной точки.

Такая блокировка может длиться существенно меньше, чем при использовании аппарата намерений, поэтому она называется короткой.

Рассмотренный способ блокировки реализован в СУБД «Набоб», ADABAS, TOTAL, IDMS. В случае неавтоматической блокировки возможен затор системы, отказ, требующий вмешательства средств восстановления (отказ будет описан ниже).

При разработке АИС одной из задач является определение необходимого уровня блокировки доступа с тем, чтобы обеспечить целостность БД. В работе [12] формально описаны различные уровни блокировки. Для этого сначала вводится следующее определение Измененные транзакцией данные становятся завершенными, когда транзакция отказывается от права аннулировать выполненные изменения, делая тем самым новые записи доступными другим транзакциям. Данные, обновляемые транзакцией, являются незавершенными, если изменения данных не завершены транзакцией. Таким образом, обновление незавершенных данных другой транзакцией может нарушить непротиворечивость БД, а чтение — получить результаты из БД с нарушенной непротиворечивостью, которые могут оказаться неправильными.

Введем определение четырем степеням непротиворечивости базы данных по отношению к заданной транзакции.

Степень 3. Для транзакции Т обеспечивается непротиворечивость БД степени 3, если выполняются следующие условия:

Т не обновляет незавершенных данных другой транзакции;

Т не разрешает доступа к измененным ею данным до завершения всех выполняемых ею изменений (т. е. до конца транзакции);

Т не читает незавершенных данных от других транзакций;

другие транзакции не изменяют данных (не делают незавершенными данные), которые читает Т, до окончания Т.

Степень 2. Для транзакции Т обеспечивается непротиворечивость БД степени 2, если выполняются первые 3 условия для степени 3.

Степень 1. Для транзакции Т обеспечивается непротиворечивость БД степени 1, если выполняются первые 2 условия для степени 3.

Степень 0. Для транзакции Т обеспечивается непротиворечивость БД степени 0, если выполняется одно первое условие для степени 3.

Одним из подходов для определения необходимого уровня блокировки СУБД (возможностей протокола блокировки) является выявление требуемых степеней непротиворечивости для всех транзакций, которые должны функционировать в системе. При формулировании требований существенное значение имеет характер взаимодействия транзакций с БД. При произвольном характере непротиворечивость БД может быть полностью гарантирована, если для всех транзакций обеспечивается непротиворечивость БД степени 3. С другой стороны, если доступ к БД регламентирован некоторыми технологическими и алгоритмическими особенностями АИС, то степень непротиворечивости может быть уменьшена в зависимости от характера взаимодействия транзакций между собой и с БД.

СУБД «Ока» обеспечивает только длинную блокировку, т. е. блокировка устанавливается автоматически на время работы транзакции. В описании прикладной программы PSB содержатся намерения программы по отношению к каждому типу сегмента, с которым она взаимодействует. По отношению к транзакции, которой соответствует эта программа, можно указать фактически четыре намерения по каждому сегменту БД: не используется (не чувствителен), только чтение, обновление, монопольное использование.

В табл. 2 приведена матрица принятия решений, на основе которой управляющая программа принимает решения по запуску очередной транзакции, находящейся в очереди, сопоставляя ее намерения и намерения выполняемых в настоящий момент транзакций по каждому типу чувствительного для планируемой транзакции сегмента. В этих условиях обеспечиваются две степени непротиворечивости:

для монопольного режима — степень 3;

для защищенного режима — степень 1.

Таким образом, невозможно в системе работать со степенью непротиворечивости, меньшей единицы.

Таблица 2

Намерения планируемой транзакции	Намерения выполняемой транзакции: монопольное использование	обновление	только чтение	сегмент не чувствителен
монопольное использование	—	—	—	+
обновление	—	—	+	+
только чтение	—	+	+	+
Режим работы	монопольный	защищенный	разделяемый	разделяемый

Примечание

— планируемая транзакция не может выполняться совместно;
+ планируемая транзакция может выполняться совместно.

Отказы. Выделяются две категории: неисправности и отказы. Неисправности связаны со сбоем или неправильной работой оборудования, программного обеспечения или человека и вносят ошибки. Эти ошибки могут привести к отказам, которые состоят во временном прерывании нормальной работы системы или выдаче неправильных выходных данных. В АИС должно иметься средство, обеспечивающее [30]:

обнаружение отказов;

сообщения об отказах при их обнаружении;

исправление причин, повлекших за собой отказ;

восстановление и повторный запуск после отказов.

В одних случаях неисправности и отказы возникают одновременно, в других неисправность может внести ошибку, которая будет существовать в течение длительного времени, прежде чем проявится во время отказа. В некоторых случаях с целью обеспечения большей скорости восстановления и повышения общей производительности АИС целесообразно проводить профилактические работы для выявления таких неисправностей до возникновения их во время отказа.

Удобной с точки зрения анализа средств восстановления является классификация отказов в зависимости от логики восстановления, т. е. к одному типу будут относиться все отказы, требующие одинаковых шагов восстановления:

затор в АИС. Несколько транзакций не могут продолжить работу из-за взаимной блокировки;

аварийное завершение транзакции;

внезапное прекращение функционирования АИС с прерыванием работы одной или нескольких транзакций;

физическое разрушение части или всего пространства памяти, после чего не могут читаться записанные там данные;

обнаружение системой нарушения непротиворечивости БД.

Рассмотрим подробнее каждый из отказов.

Затор в АИС. Блокировка элемента данных при работе транзакции может вызвать установку всей системы в ожидание завершения некоторых событий, завершение которых невозможно без специального вмешательства. Например, транзакция А блокирует запись 1, транзакция В — запись 2. А выдает команду чтения записи 2 с намерением обновления, В — записи 1 с намерением обновления.

Обе транзакции не смогут сдвинуться с места. Такая ситуация часто называется «смертельными объятиями» (*deadly embrace*).

Аварийное завершение транзакции. Этот отказ может возникнуть, когда система прекращает выполнение транзакции из-за неправильной работы программы (выход за границы раздела, переполнение и т. д.). Ошибка может быть транзитивной, например из-за сбоя процессора или неверной информации на входе, или постоянной.

Внезапное прекращение работы системы. Этот отказ может возникнуть из-за сбоя центрального процессора, «зависания» операционной системы или СУБД, ошибки в транзакции, которая приводит к нарушению работы системного программного обеспечения, переполнению очередей входных сообщений или области памяти БД и т. д.

Физическое разрушение памяти. ЭВМ не может считать одну или несколько записей с диска либо в результате физического повреждения диска (соприкосновение головок с поверхностью), либо в результате внешних воздействий, которые приводят к частичному разрушению информации (ошибка при контрольном суммировании). Этот отказ фиксируется аппаратно или операционной системой.

Обнаружение системой нарушения непротиворечивости БД. Этот вид отказа характерен тем, что с точки зрения операционной системы и аппаратных средств в БД хранится правильная информация, но при обращении транзакции в БД СУБД при разборе записей обнаруживает нарушение непротиворечивости, которое может выражаться либо в отсутствии необходимых связей, либо в нарушении структуры физической организации записей.

Последний тип отказа может проявляться значительно позже возникновения неисправности, нарушившей непротиворечивость БД, так как ошибка обнаруживается только при обращении СУБД к испорченному элементу данных. Если не будут предусмотрены специальные средства проверки непротиворечивости БД, эта ошибка может быть не обнаружена в течение длительного времени и вызвать серьезные затруднения по ее устранению.

Действия и процедуры восстановления

Для каждого типа отказа должна быть определена связанная с ним процедура восстановления и повторного запуска АИС, кото-

рая представляет собой набор заранее определенных шагов, обеспечивающих возврат системы в состояние, предшествующее появлению отказа. При этом должна быть обнаружена и устранена причина отказа. Выполнение процедур восстановления возможно при регулярном выполнении в процессе функционирования АИС соответствующих действий по поддерживанию различных видов избыточных данных. Выполняемые действия, структура и состав формируемых ими избыточных данных, алгоритмы процедур восстановления существенно зависят от используемого аппарата управления данными, в первую очередь от физической организации хранимых данных и способов обновления данных. С другой стороны, при разработке аппарата управления данными должны учитываться требования по обеспечению целостности. Дальнейшие рассуждения проводятся исходя из следующих предположений:

1) в АИС для всех транзакций обеспечивается непротиворечивость БД степени не ниже 1, т. е. элементы данных, изменяемые одной транзакцией, не могут изменяться другой до завершения изменения первой;

2) затор в АИС и аварийное завершение транзакций устраняются одинаковым способом, т. е. затор рассматривается как транзитивная ошибка в программе;

3) массивы могут располагаться как на отдельных устройствах, так и разделять одно устройство, имеют границу и могут восстанавливаться при отказах независимо друг от друга;

4) поддерживается одновременно одна версия массива, находящегося в точном состоянии, т. е. не рассматривается метод ведения параллельно нескольких копий всего массива, который используется, например, в системе управления полетами космических кораблей и рассмотрен в [21] в качестве одного из способов обеспечения целостности АИС;

5) отказы — аварийное завершение транзакции — происходят из-за транзитивных ошибок.

Тогда существует конечное множество отказов $E=\{e_i\}$, где $i=1, 2, 3, 4$, которые могут возникнуть во время функцинирования АИС:

e_1 — аварийное завершение транзакции;

e_2 — внезапное прекращение работы системы;

e_3 — физическое разрушение памяти;

e_4 — обнаружение системой нарушения непротиворечивости БД.

Множеству E однозначно соответствует множество классов процедур восстановления $P=\{p_i\}$, причем использование при отказе e_i класса процедур p_i приведет к восстановлению АИС в точное состояние. Каждый класс процедур p_i представляет собой конечное множество (которое является подмножеством множества P) процедур $p_i=\{p_{ij}\}$, $j=1, 2, \ldots, n$, причем каждая p_{ij} обеспечивает восстановление БД при отказе e_i в точное состояние. Класс процедур восстановления p_i определяет, что должна сделать процедура восстановления при отказе e_i, а конкретная процедура p_{ij} оп-

ределяет, как это можно сделать в рамках определенного аппарата управления данными.

Введем четыре типа точек синхронизации целостности (ТСЦ) АИС:

s_1 — фиксация транзакции (начало выполнения транзакции или контрольная точка транзакции);

s_2 — фиксация системы (например, контрольная точка системы);

s_3 — фиксация БД, свободной от ошибок, влекущих за собой отказ e_3;

s_4 — фиксация БД, свободной от ошибок, влекущих за собой отказ e_4.

Будем предполагать, что средства восстановления АИС обеспечивают возможность перехода в эти состояния. Определим, что формально соответствует классам процедур, основываясь на ранее введенных определениях условий нарушения непротиворечивости:

p_1 — аннулирование изменений аварийно завершенной транзакции, продолжение выполнения других, повторение прерванной транзакции (для транзитивных ошибок) с ее ТСЦ s_1;

p_2 — аннулирование изменений всех прерванных транзакций, перевод АИС в ТСЦ s_2, перевод АИС в ТСЦ s_1 для первой начавшейся из всех прерванных транзакций и повторное выполнение всех прерванных транзакций;

p_3 — аннулирование изменений всех прерванных транзакций, взаимодействовавших с разрушенным массивом (массивами), перевод разрушенного массива (массивов) в состояние ТСЦ s_3, а затем в состояние ТСЦ s_2 и ТСЦ s_1 для первой начавшейся из прерванных транзакций, повторное выполнение всех прерванных транзакций;

p_4 — аннулирование изменений всех прерванных транзакций, взаимодействовавших с разрушенным массивом (массивами), перевод разрушенного массива (массивов) в состояние ТСЦ s_4, а затем ТСЦ s_2 и ТСЦ s_1 для первой начавшейся из прерванных транзакций, повторное выполнение всех прерванных транзакций.

Наиболее распространенными действиями для обеспечения точек синхронизации целостности являются [7, 8, 21]:

ведение журнала изменений БД и сообщений;

выполнение контрольной точки;

выполнение дампа БД;

выполнение ревизирования БД и в случае отсутствия ошибок выполнение дампа.

В большинстве СУБД журнал ведется на магнитной ленте, в некоторых, например SYSTEM 2000 [8], журнал располагается на дисках. Некоторые системы для повышения надежности восстановления позволяют вести журнал одновременно в двух экземплярах, например IMS/VS [12]. Записи в журнале могут быть блокированные и неблокированные. При блокированных записях сокращаются накладные расходы на ведение журнала и повышается