Лекции 8-9. Методы повышения надежности программ и оценка эффективности их применения.

1. Влияние избыточности на повышение надежности программ

Так как в программах нет необходимости "ремонта" компонент с участием человека, то можно добиваться высокой автоматизации программного восстановления. Главной задачей становится восстановление за время, не превышающее порогового значения между сбоем и отказом. Автоматизируя процесс и сокращая время восстановления, можно преобразовать отказы в сбои и тем самым улучшить показатели надежности функционирования системы.

Комплексы программ в процессе функционирования находятся под воздействием *шумов-возмущений* различных типов. Для снижения их влияния на результаты следует применять *фильтры*, позволяющие обнаруживать искажения, устранять или уменьшать их вредные последствия. Разнообразие видов искажений приводит к необходимости построения систем фильтров, каждый из которых способен селектировать некоторые виды искажений.

Для реализации фильтров, обеспечивающих повышение надежности функционирования программ и защиту вычислительного процесса и информации программно-алгоритмическими методами, используется программная, информационная и временная *избыточность*. Основная задача ввода избыточности состоит в исключении возможности аварийных последствий от возмущений, соответствующих отказу системы. Любые аномалии при исполнении программ необходимо сводить до уровня сбоя путем быстрого восстановления.

Временная избыточность состоит в использовании некоторой части производительности ЭВМ для контроля исполнения программ и восстановления вычислительного процесса. Величина временной избыточности зависит от требований к надежности функционирования системы и находится в пределах от 5-10% производительности однопроцессорной ЭВМ до трех-четырех кратного дублирования производительности машины. Временная избыточность используется на обнаружение искажений, их диагностику и на реализацию операций восстановления. На это требуется в общем случае небольшой интервал времени, который выделяется либо за счет резерва, либо за счет сокращения времени решения функциональных задач.

Информационная избыточность состоит в дублировании накопленных исходных и промежуточных данных, обрабатываемых комплексом программ. Избыточность используется для сохранения достоверности данных, которые в наибольшей степени влияют на нормальное функционирование программ или требуют значительного времени восстановления. Для менее важных данных информационная избыточность используется в виде помехозащитных кодов, позволяющих только обнаружить искажение.

Программная избыточность используется для контроля и обеспечения достоверности наиболее важных решений по управлению и обработке информации. Она заключается в применении нескольких вариантов программ, различающихся методами решения задачи или программной реализацией одного и того же метода. Программная избыточность необходима также для реализации программ контроля и восстановления данных с использованием информационной избыточности и для функционирования всех средств защит, использующих временную избыточность.

2. Эффективность применения избыточности для повышения надежности комплексов программ.

Выше рассмотрены принципы использования временной, информационной и программной избыточности для обеспечения надежности программ. Здесь представлены некоторые рекомендации, позволяющие оценить эффективность использования избыточности. Наибольшее внимание уделяется временной избыточности, которая эффективно используется для оперативной защиты при отказовых ситуациях и непосредственно влияет на надежность функционирования программ.

Для реализации стратегий резервирования программ необходима *временная избыточность*. При этом временная избыточность используется в основном для оперативного контроля состояния данных и вычислительного процесса, а также для автоматического восстановления при возникновении отказовых ситуаций. Резерв времени для выполнения этих операций можно считать достаточным независимо от числа предыдущих отказов, времени на их устранение и наработки на отказ. Кроме того, суммарное время восстановления работоспособности обычно не ограничено.

Эффективность оперативного использования временной избыточности для повышения надежности функционирования программ определяется затратами на контрольно-восстановительные операции, изменением показателей надежности в зависимости от затрат и связью этих изменений с отлаженностью программ. В результате для оценки эффективности введения временной избыточности в программе необходимо:

* определить совокупные затраты на контроль, на работу при необнаруженном искажении и на восстановление, обеспечивающие заданную вероятность обнаружения отказовой ситуации при исполнении программ ;
* определить основные показатели надежности функционирования программ в зависимости от совокупных затрат на оперативный контроль и восстановление;
* оптимизировать суммарные затраты на отладку программ и оперативную защиту от искажений для обеспечения заданной надежности функционирования комплекса программ.

При решении последней задачи источниками искажений предполагаются только не выявленные ошибки в программах.

3. Влияние оперативного контроля и восстановления на производительность ЭВМ.

Использование времени функционирования ЭВМ для контроля работоспособности, исправления искажений и восстановления при отказовых ситуациях приводит к снижению затрат производительности ЭВМ на исполнение комплекса программ в процессе эксплуатации. В результате сокращаются ресурсы ЭВМ, доступные для выполнения основных функций системы. Это сокращение ресурсов можно отразить *коэффициентом простоя* ЭВМ $\mathbf{К_п = 1 - К_г}$, характеризующим относительные затраты времени на задачи повышения помехозащищенности программ. По мере совершенствования и углубления средств помехозащиты программ возрастают затраты времени ЭВМ на их исполнение, что отражается на снижении реальной эффективности функционирования комплекса программ.

Целесообразно исследовать влияние использования временной избыточности на сокращение полезной производительности ЭВМ для решения функциональных задач, а также на снижение потерь вследствие отказов. После определения этих составляющих можно провести их совместный анализ для оптимизации глубины помехозащиты программ с учетом затрат на ее реализацию.

4. Методы обеспечения надежности программ. Основные факторы, определяющие надежность программ.

Надежность функционирования комплексов программ зависит от многих факторов, которые можно объединить в три группы (табл.1).

- факторы,непосредственно определяющие возникновение отказов, и характеристики надежности программ;
- методы, способствующие проектированию корректных программ, снижающие вероятность программных ошибок и отказовых ситуаций при исполнении программ;
- методы, активно влияющие на повышение надежности функционирования программ позволяющие контролировать и поддерживать заданные показатели надежности.

Таблица 1.

Факторы, непосредственно определяющие надежность программ	Методы проектирования корректных программ	Методы контроля и обеспечения надежности программ
Особенности внешних абонентов и пользователей результатов программ Требования к показателям надежности Инерционность внешних абонентов Необходимое время отклика на входные данные	**Структурное проектирование программ и данных** Структурное проектирование программных модулей Структурное проектирование взаимодействия модулей Структурирование данных	**Методы использования избыточности** Временной Информационной Программной **Методы контроля программ, данных и вычислительного процесса** Предпусковой контроль Оперативный контроль
Искажения исходных данных Искажения данных поступающих от человека Искажения данных поступающих по телекодовым каналам связи Искажения данных в процессе накопления и хранения в ЭВМ	**Тестирование программ** Детерминированное тестирование Статистическое тестирование Динамическое тестирование и контроль пропускной способности в реальном времени	**Методы программного восстановления** Восстановление текстов программ Исправление данных Корректировка вычислительного процесса **Методы испытаний на надежность** Форсированные испытания Испытания в нормальных условиях эксплуатации Расчетно-экспериментальные методы определения надежности
Ошибки в программах и их проявление Статистические характеристики ошибок и искажений: Программ массивов данных Вычислительного процесса		**Методы обеспечения надежности при сопровождении** Обеспечение сохранности программ эталонных версий Обеспечение корректности внесения изменений и развития версий

Отказы при функционировании программ возникают вследствие взаимодействия данных и ошибок в программе. Характеристики ошибок в значительной степени определяются методами проектирования программ.Активные методы обнаружения и локализации программных ошибок базируются на тестировании различных видов: детерминированном, статистическом и динамическом. Однако все виды тестирования не гарантируют отсутствие ошибок в комплексах программ и высокую надежность. Для

этого применяются методы контроля и обеспечения надежности программ путем использования программной, временной и информационной избыточности.

Ситуации проявления ошибок при исполнении программ можно разделить на три группы:

- *Отказ* - искажения вычислительного процесса, данных или программ, вызывающие полное прекращение выполнения функций системой;
- Искажения, кратковременно прерывающие функционирование системы - *отказовые ситуации* или *сбои*, характеризующиеся быстрым восстановлением без длительной потери работоспособности;
- Искажения, мало отражающиеся на вычислительном процессе - *сбои* и *искажения*, не создающие отказовые ситуации.

Основным показателем при классификации конкретных видов искажений является возможное время восстановления и степень проявления последствий произошедшего сбоя или отказовой ситуации:

- зацикливание, т.е. последовательная повторяющаяся реализация некоторой группы команд, не прекращающаяся без внешнего (для данного цикла) вмешательства;
- останов исполнения программ и прекращение решения функциональных задач;
- полная самоблокировка вычислительного процесса (клинч) из-за последовательного перекрестного обращения разных программ к одним и тем же ресурсам ЭВМ без освобождения ранее занятых ресурсов;
- прекращение или значительное снижение темпа решения некоторых задач вследствие перегрузки ЭВМ по пропускной способности;
- значительное искажение или полная потеря накопленной информации о текущем состоянии управляемого процесса или внешних абонентов;
- искажение процессов взаимного прерывания программ, приводящее к блокировке возможности некоторых типов прерываний.

5. Методы программного восстановления.

Выбор метода оперативного восстановления происходит в условиях неопределенности сведений о характере отказовой ситуации и степени ее влияния на работоспособность программ.

Каждый метод восстановления характеризуется следующими статическими параметрами:

- вероятность полного восстановления нормального функционирования комплекса программ при данном методе (p_3);
- затратами ресурсов ЭВМ на проведение процедуры восстановительных работ выбранным методом (b_3);
- длительностью проведения работ по восстановлению - суммарным временем выбора метода восстановления и временем его реализации(t_3)

Показатели восстановления p_3 и t_3 непосредственно влияют на показатели надежности функционирования комплекса программ. Если операции по восстановлению работоспособности комплекса программ при отказовой ситуации полностью завершаются за время меньше $t_д$ и после этого продолжается нормальное функционирование, то происшедшее искажение в работе программ не учитывается как отказ и не влияет на основные показатели надежности.

Процесс функционирования комплекса программ на однопроцессорной ЭВМ в реальном масштабе времени с учетом операций контроля и восстановления можно

представить графом состояний, дуги которого соответствуют возможным переходам между состояниями за некоторый интервал времени.

Основные состояния следующие:

0- состояние соответствует нормальному функционированию работоспособного комплекса программ при полном отсутствии искажений - *полезная работа;*

1- состояние имеет место при переходе комплекса программ в режим контроля функционирования и обнаружения ошибок - *состояние контроля;*

2- состояние соответствует функционированию программ при наличии искажений, не обнаруженных средствами контроля - *состояние необнаруженного искажения* данных или вычислительного процесса, которое, в частности может соответствовать отказу;

3- состояние характеризуется функционированием группы программ восстановления режима полезной работы и устранения последствий искажения - *восстановление после действительного искажения;*

4- состояние соответствует также восстановлению режима полезной работы, но после ложного обнаружения проявления искажения, когда в действительности состояние полезной работы не нарушалось - *восстановление после ложной тревоги.*

Переходы между состояниями могут происходить в некоторых направлениях случайно или коррелированно с предыдущем переходом (рис.1). Пребывание во всех состояниях, кроме нулевого, сопряжено с затратами производительности ЭВМ на выполнение операций, не связанных с прямыми функциональными задачами, и может рассматриваться как снижение общей эффективности комплекса программ и производительности ЭВМ. При определении показателей надежности учитывается только такая цепочка последовательных состояний вне работоспособности, которая оказывается протяженности больше $t_д$. Все остальные более короткие выходы из нулевого состояния не влияют на показатели надежности.

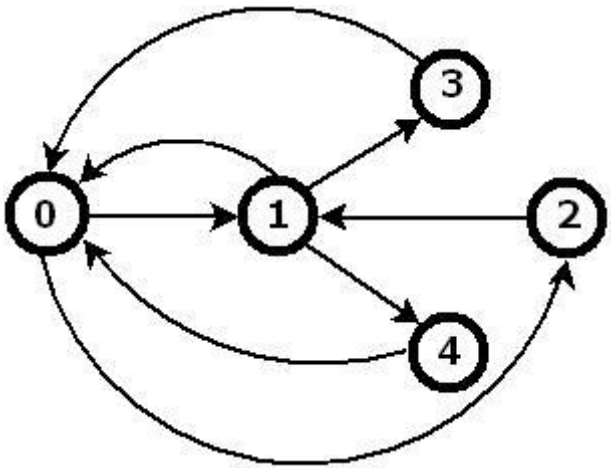

Рис. 1. Граф переходов между состояниями.

6. Методы испытаний программ на надежность.

В теории надежности разработан ряд методов, позволяющих определить характеристики надежности сложных систем. Эти методы можно свести к трем основным группам:

- прямые экспериментальные методы определения показателей надежности систем в условиях нормального функционирования;
- форсированные методы испытаний реальных систем на надежность;
- расчетно-экспериментальные методы, при использовании которых ряд исходных данных для компонент получается экспериментально, а окончательные показатели надежности систем надежности рассчитываются с использованием этих данных.

Прямые экспериментальные методы определения показателей надежности программ в нормальных условиях функционирования в ряде случаев трудно использовать из-за больших значений времени наработки на отказ (сотни и тысячи часов).

Форсированные методы испытаний надежности программ значительно отличаются от традиционных методов испытаний аппаратуры. Форсирование испытаний может выполняться путем повышения интенсивности искажений исходных данных, а также специальным увеличением загрузки комплекса программ выше нормальной.

Особым видом форсированных испытаний является проверка эффективности средств контроля и восстановления программ, данных и вычислительного процесса. Для этого имитируются запланированные экстремальные условия функционирования программ, при которых в наибольшей степени стимулируется работа испытываемого средства программного контроля или восстановления.

Расчетно-экспериментальные методы . При анализе надежности программ применение расчетно-экспериментальных методов более ограничено, чем при анализе аппаратуры. Это обусловлено неоднородностью надежностных характеристик основных компонент: программных модулей, групп программ, массивов данных и т.д. Однако в некоторых случаях расчетным путем можно оценить характеристики надежности комплексов программ. Сочетание экспериментальных и аналитических методов применяется также для определения пропускной способности комплекса программ на конкретной ЭВМ и влияние перегрузки на надежность его функционирования.

7. Методы обеспечения надежности комплексов программ при сопровождении.

В этой стадии жизненного цикла программ расширяются условия их использования и характеристики исходных данных, вследствие чего могут потребоваться изменения в программах. Для сохранения и улучшения показателей надежности комплексов программ в процессе длительного сопровождения необходимо четко регламентировать передачу комплексов программ пользователям. Целесообразно накапливать необходимые изменения в программах и вводить их группами, формируя *очередную версию* комплекса программ с измененными характеристиками. Версии комплекса программ можно разделить на эталонные и пользовательские (или конкретного объекта).

Эталонные версии развиваются, дорабатываются и модернизируются основными разработчиками комплекса программ или специалистами, выделенными для их сопровождения. Они снабжаются откорректированной технической документацией,

полностью соответствующей программам, и точным перечнем всех изменений введенных в данную версию по сравнению с предыдущей.

Пользовательские версии . Необходимы также общие проверки работоспособности и сохранности всех программ комплекса. Для корректности выполнения изменений они снабжаются методиками проверки и правилами подготовки контролирующих тестов. . Целесообразно ограничивать доступ широких пользователей к технологической документации, хранящей подробные сведения о содержании и логике функционирования программ. Такие меры в некоторой степени предотвращают возможность резкого ухудшения показателей надежности.