ДЕЦЕНТРАЛИЗОВАННЫЙ АЛГОРИТМ САМОДИАГНОСТИКИ РАСПРЕДЕЛЁННЫХ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ[1]

Молдованова Ольга Владимировна
ФГОБУ ВПО «СибГУТИ», г. Новосибирск
E-mail: ovm@csc.sibsutis.ru Тел. (383) 269-82-75

1. Введение

Распределённые вычислительные системы (ВС) являются важнейшим инструментом решения сложных научных, инженерных и экономических задач [1]. Основным функциональным элементом распределённой ВС является элементарная машина (ЭМ). Такие системы характеризуются большемасштабностью – количество ЭМ в их составе может достигать $10^5 – 10^6$. Несмотря на высокую надёжность микроэлектронной базы, вероятность возникновения отказов в распределённых ВС повышается с ростом количества элементарных машин. При этом в последнее время постоянно увеличивается число трудоёмких задач, решаемых на большемасштабных ВС. Следовательно, организация отказоустойчивого функционирования таких систем требует создания алгоритмических и программных средств самоконтроля и самодиагностики.

В течение нескольких десятилетий было предложено немало методик диагностики отказов в вычислительных системах [2–7]. Большинство работ базируется на стратегии, предложенной в [2], согласно которой ЭМ способны тестировать друг друга. При этом исправные ЭМ могут безошибочно определить состояние тестируемых ими элементарных машин. А результат тестирования неисправными ЭМ может быть произвольным. Все результаты тестов собираются на высоконадёжной управляющей элементарной машине (центральном обозревателе), которая и определяет диагностический образ системы.

Опыт разработок в области самодиагностики большемасштабных распределённых вычислительных систем показывает, что централизованный подход ведёт к снижению производительности ВС и нарушает важный принцип их построения: отказ одной ЭМ влечёт за собой отказ всей системы. Эти проблемы могут быть решены при децентрализации процесса диагностирования.

Методология децентрализованной самодиагностики была сформулирована в работах [3, 4]. Её основой является предположение, что каждая исправная ЭМ может определить корректный диагностический образ всей системы, базируясь на результатах взаимных тестов других исправных элементарных машин этой ВС. Таким образом, передача тестовой информации осуществляется только между исправными компонентами системы, что гарантирует изоляцию неисправных и препятствует их влиянию на формирование диагностического образа ВС. В дальнейшем эта идея получила развитие в работах других исследователей [5–7].

Основными недостатками ранее предложенных алгоритмов являются:

- ограничение на тестовую топологию (древовидная тестовая топология);
- изменение состояния ЭМ не может происходить во время фазы тестирования;
- последовательная передача диагностических сообщений.

В докладе предлагается децентрализованный алгоритм самодиагностики распределённых ВС, обладающий следующими характеристиками:

- каждая исправная ЭМ тестируется только одной другой машиной;
- передача диагностической информации происходит только при изменении состояния тестируемой ЭМ;
- изменение состояния ЭМ может происходить во время фазы тестирования;

[1] Работа выполнена при поддержке Совета по грантам Президента РФ (ведущая научная школа НШ 5176.2010.9) и Российского фонда фундаментальных исследований (гранты 11-07-00109-а, 09-07-00095-а).

– фазы тестирования и распространения диагностической информации выполняются параллельно.

2. Постановка задачи

Рассматривается распределённая вычислительная система, состоящая из N элементарных машин (в дальнейшем узлов), соединённых друг с другом каналами связи.

Каждый узел может находиться в исправном или неисправном состоянии. Исправный узел обладает информацией о том, какие узлы являются его соседями. Кроме того, он способен инициировать тестирование соседнего узла и отвечать на тестовые запросы своих соседей. В качестве теста используются диагностические сообщения вида «Are you alive?». Исправный узел всегда отвечает на тестовый запрос в течение определённого периода времени (тайм-аута), способен передавать диагностическую информацию своим соседям и запрашивать их стать его тестерами.

Узел, находящийся в неисправном состоянии, не способен отвечать на любые сообщения от своих соседей, передавать им диагностическую информацию или запросы стать его тестерами. Таким образом, узлы в системе не могут информировать друг друга о своей неисправности.

Если в течение тайм-аута ответ на тестовый запрос от узла не получен, то узел-тестер делает заключение о неисправности тестируемого им узла. Величина тайм-аута определяется как функция задержки каналов связи.

Узел распределённой ВС в любой момент времени может перейти в неисправное состояние. В свою очередь неисправный узел может быть отремонтирован и снова введён в эксплуатацию. При этом узел получает всю необходимую информацию, касающуюся его соседей, но не обладает информацией о текущем диагностическом образе системы.

Контроль и диагностика неисправностей каналов связи алгоритмом не предусматривается. Поэтому не делается различия между неисправностью тестируемого узла и канала связи, соединяющего его с тестером.

3. Описание алгоритма

Алгоритмом предусматривается, что узлы обнаруживают изменение состояния соседних с ними узлов, а затем передают эту диагностическую информацию другим узлам системы. Диагностическая информация состоит из событий двух видов: переход узла из исправного состояния в неисправное и наоборот.

Обнаружение изменения состояния узлов в вычислительной системе осуществляется путем их периодического тестирования (сообщение типа «TestReq»). Каждый узел тестируется только одним исправным узлом. Если в течение определённого тайм-аута ответ (сообщение типа «TestResp») получен, узел диагностируется как исправный, иначе – как неисправный. Сразу после диагностирования изменения состояния тестирующий узел начинает этап передачи диагностической информации (сообщение типа «NewEvent») своим соседям, а те в свою очередь своим и т.д.

После того как узел i отправил диагностическое сообщение соседнему с ним узлу j, он ожидает от j подтверждения (сообщение типа «AckNewEvent») получения сообщения в течение определённого тайм-аута. Если такое подтверждение не поступает, узел i начинает новый этап распространения информации о неисправности узла j. Таким образом, удаётся получить сведения об изменении состояния узлов, не дожидаясь следующего раунда тестирования.

Если узел j исправен, то после получения диагностического сообщения от i он проверит, является ли информация в этом сообщении новой, старой или такой же, какой обладает и он.

Пусть t_j – время определения диагностического образа системы узлом j, а t_i – узлом i. Если $t_j = t_i$, то узлы i и j имеют одинаковые данные о состоянии всех узлов в ВС. И узел j не передаёт полученное им сообщение от i далее своим соседям.

Если $t_j > t_i$, то узел j обладает более новой информацией о состоянии хотя бы одного из узлов распределённой ВС. В этом случае j передаёт i свои данные о диагностическом образе вычислительной системы.

Если $t_j < t_i$, то j содержит более старую информацию о состоянии хотя бы одного из узлов системы. В таком случае j обновляет свои данные и передаёт их далее своим соседям.

В результате выполнения описанного выше алгоритма тестирования один или несколько узлов в системе могут быть «брошены», т.е. они перестают тестироваться другими узлами.

Все неисправные узлы являются «брошенными», поскольку после диагностирования их неисправности узел-тестер перестаёт их тестировать. Только после ремонта и нового ввода в эксплуатацию, т.е. изменения состояния на исправное, такие узлы будут снова тестироваться. Восстановленный узел обращается ко всем своим соседям по очереди с запросом на тестирование (сообщение типа «TestMeRepair»), пока не получит сообщение, подтверждающее согласие стать его тестером (сообщение типа «AckTestMeRepair»). После этого узел-тестер начинает раунд тестирования. Кроме этого, вновь исправный узел передаёт всем своим соседям информацию о своей исправности (сообщение типа «Repair»).

Исправный узел также может стать «брошенным», в случае если его узел-тестер поменял своё состояние на неисправное. Так же как и в случае с неисправным «брошенным» узлом, он должен обратиться ко всем своим соседям по очереди с запросом на тестирование (сообщение типа «TestMe»).

На рис. 1 приведена диаграмма состояний, описывающая поведение узла распределённой ВС, при выполнении децентрализованного алгоритма самодиагностики.

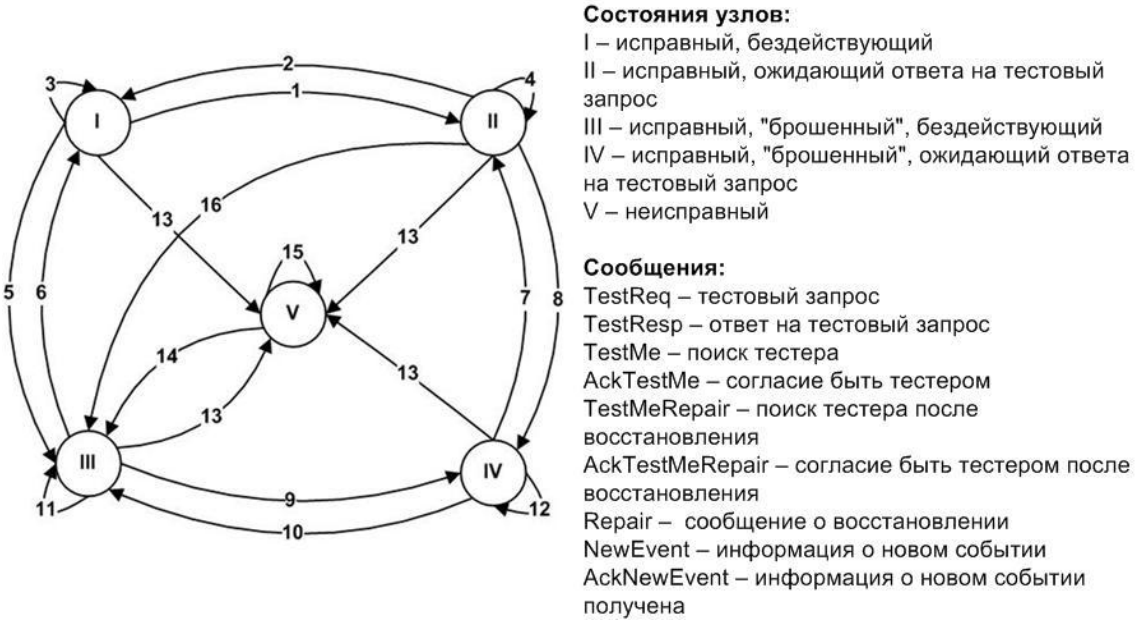

Рис. 1. Диаграмма состояний узла при выполнении децентрализованного алгоритма самодиагностики распределённой ВС.

В табл. 1 приведена информация о событиях, которые приводят к переходам из состояния в состояние при выполнении описанного выше алгоритма.

Для хранения и сбора диагностической информации в ходе выполнения алгоритма самодиагностики на каждом узле j вычислительной системы используются следующие структуры данных:

– массив $events_j[N]$ счётчиков событий, где N – количество узлов в диагностируемой системе. Если $events_j[i]$ – чётное число, то узел i исправен, иначе – не исправен. Начальное значение для элементов массива – 0;

– массив $testnbs_j[N]$, где N – количество узлов в диагностируемой системе; $testnbs_j[i] = 0$, если узлы i и j не являются соседями; $testnbs_j[i] = 1$, если узел i тестируется узлом j; $testnbs_j[i] = 2$, если узел i тестирует узел j; $testnbs_j[i] = 3$, если узлы i и j являются соседями, но не тестируют друг друга; $testnbs_j[i] = 4$, если узлы i и j тестируют друг друга.

Таблица 1. Описание событий для переходов диаграммы состояний.

Номер перехода	**События**
1	Послано сообщение TestReq
2	Получено сообщение TestResp Сообщение TestResp не получено в течение тайм-аута Получено сообщение TestMeRepair от тестируемого узла Получено сообщение Repair от тестируемого узла
3	Получено сообщение TestReq Получено сообщение NewEvent Получено сообщение TestMe Получено сообщение AckNewEvent Получено сообщение TestMeRepair Получено сообщение Repair Не получено сообщение AckNewEvent в течение тайм-аута
4	Получено сообщение TestReq Получено сообщение NewEvent Получено сообщение TestMe Получено сообщение AckNewEvent Получено сообщение TestMeRepair Получено сообщение Repair Не получено сообщение AckNewEvent в течение тайм-аута
5	Получено сообщение NewEvent Получено сообщение TestMe от тестера Получено сообщение TestMeRepair от тестера Не получено сообщение AckNewEvent в течение тайм-аута от тестера
6	Получено сообщение AckTestMe Получено сообщение NewEvent Получено сообщение AckTestMeRepair
7	Получено сообщение AckTestMe Получено сообщение AckTestMeRepair
8	Получено сообщение NewEvent Получено сообщение TestMe от тестера Получено сообщение TestMeRepair от тестера Не получено сообщение AckNewEvent в течение тайм-аута от тестера
9	Послано сообщение TestReq
10	Получено сообщение TestResp Сообщение TestResp не получено в течение тайм-аута Получено сообщение TestMeRepair от тестируемого узла Получено сообщение Repair от тестируемого узла
11	Не получено сообщение AckTestMe в течение тайм-аута Получено сообщение NewEvent Получено сообщение TestMe

Номер перехода	События
	Получено сообщение AckNewEvent Получено сообщение TestMeRepair Получено сообщение Repair Не получено сообщение AckTestMeRepair в течение тайм-аута Получено сообщение TestReq Не получено сообщение AckNewEvent в течение тайм-аута
12	Получено сообщение NewEvent Получено сообщение TestMe Получено сообщение AckNewEvent Получено сообщение TestMeRepair Получено сообщение Repair Не получено сообщение AckTestMe в течение тайм-аута Получено сообщение TestReq Не получено сообщение AckTestMeRepair в течение тайм-аута Не получено сообщение AckNewEvent в течение тайм-аута
13	Узел не исправен
14	Узел восстановлен
15	Узел не исправен
16	Не получено сообщение TestResp в течение тайм-аута

4. Результаты моделирования

Моделирование предложенного алгоритма проводилось с использованием средства дискретного моделирования YACSIM [8]. Это событийно- и процессно-ориентированный инструментарий моделирования, позволяющий создавать взаимодействующие друг с другом процессы.

Программная реализация децентрализованного алгоритма самодиагностики распределённой ВС включает в себя следующие процессы:

- процесс Init, выполняющий начальную инициализацию и создающий процессы MsgHandler и Lost;
- процесс MsgHandler, отвечающий за получение и обработку всех видов сообщений (см. рис. 1);
- процесс Lost, выполняющий рассылку сообщений «TestMe» для поиска соседнего узла-тестера;
- процесс Test, реализующий рассылку тестовых запросов;
- процесс Event, выполняющий сравнение присылаемой диагностической информации с имеющейся на узле;
- процесс SendEvent, рассылающий диагностическую информацию соседним узлам.

Каждый узел ВС переключался между диагностическим алгоритмом и обычной рабочей нагрузкой. Время выполнения рабочей нагрузки на узле являлось экспоненциально распределённой случайной величиной со средним значением, равным 1 единице времени. По истечении этого времени сразу же выполнялся повторный запрос на использование процессора для выполнения рабочей нагрузки. Эти запросы обрабатывались в порядке очереди FIFO. Фоновые процессы, реализующий алгоритм самодиагностики, осуществляли запросы процессора через ту же самую очередь FIFO. Общее время моделирования составляло 1000 единиц времени.

Следующие виды задержек использовались при моделировании предложенного алгоритма:

- время формирования тестового запроса (2 единицы времени);
- время формирования ответа на тестовый запрос или подтверждающего сообщения (1 единица времени);
- время обработки ответа на тестовый запрос (1 единица времени);
- время обработки диагностического сообщения и обновления информации на узле (2,5 единиц времени);
- время определения номера соседнего уза для отправки ему диагностического сообщения (0,1 единицы времени);
- время формирования диагностического сообщения (2,5 единиц времени);
- время, затрачиваемое на пересылку любого сообщения от одного узла другому (1 единица времени);
- интервал между тестами (500 единиц времени).

Исследование разработанного алгоритма самодиагностики распределённых вычислительных систем проводилось для топологий: двумерная решётка (4 х 4), двумерный тор (4 х 4), четырёхмерный гиперкуб и трёхмерная решётка (4 х 4 х 4). Моделировалась ситуация, когда узел 1 переходил в неисправное состояние в момент времени 10 при первом раунде тестирования. Моделирование проводилось со 100 различными вариантами генерирования случайных величин. В табл. 2 приведены средние значения полученных оценок. Проанализировав результаты можно сделать вывод, что увеличение размерности топологии системы не приводит к большому увеличению времени информирования всех узлов ВС о текущем диагностическом образе.

Таблица 2. Результаты моделирования.

	Двумерная решётка (4 х 4)	Двумерный тор (4 х 4)	Четырёхмерный гиперкуб	Трёхмерная решётка (4 х 4 х 4)
Обнаружение неисправности	12,012	12,012	12,012	11,2
Время, когда последний узел системы узнаёт о неисправности	87,405	62,997	72,24	123,69
Количество передаваемых диагностических сообщений	27	37	41	220

Для топологии трёхмерная решётка (4 х 4 х 4) получены также оценки времени выполнения предложенного алгоритма в единицах времени и в процентном соотношении от общего времени моделирования (табл. 3). Результаты показывают, что загрузка системы при выполнении алгоритма в отсутствии неисправностей очень мала, и она увеличивается незначительно при наличии одной неисправности.

Таблица 3. Время выполнения алгоритма для топологии трёхмерная решётка (4 х 4 х 4).

Время выполнения алгоритма в отсутствии неисправностей	15,18 (1,518%)
Время выполнения при одной неисправности	70,5 (7,05%)

5. Заключение

В докладе предложен децентрализованный алгоритм самодиагностики распределённых ВС, характеризующийся параллельным выполнением фаз тестирования и распространения диагностической информации.

Проведено моделирование разработанного алгоритма с использованием средства дискретного моделирования YACSIM. Исследование проводилось для распространённых топологий распределённых вычислительных систем. Результаты моделирования показывают, что загрузка системы при выполнении алгоритма в отсутствии неисправностей очень мала, и она увеличивается незначительно при наличии одной неисправности.

В дальнейшем планируется провести исследования алгоритма при наличии нескольких событий, в том числе событий восстановления узла после отказа, и реализовать предложенный алгоритм как часть системного программного обеспечения самодиагностики пространственно-распределённой мультикластерной вычислительной системы Центра параллельных вычислительных технологий ФГОБУ ВПО «Сибирский государственный университет телекоммуникаций и информатики» и Института физики полупроводников им. А.В. Ржанова СО РАН.

6. Литература

1. Хорошевский В.Г. Архитектура вычислительных систем. – М.: МГТУ им. Н.Э. Баумана, 2008. – 520 с.
2. Preparata F.P., Metze G., Chien R.T. On the connection assignment problem of diagnosable systems // IEEE Trans. Electron. Comput. Dec. 1967. – vol. EC-16. – no. 6. – pp. 848–854.
3. Евреинов Э.В., Хорошевский В.Г. Однородные вычислительные системы. – Новосибирск: Наука, 1978. – 319 с.
4. Kuhl J.G., Reddy S.M. Fault-diagnosis in fully distributed systems // Proc. 11th Int. Symp. Fault-Tolerant Computing, June 1981. – pp. 100–105.
5. Bagchi A., Hakimi S.L. An optimal algorithm for distributed system level diagnosis // Proc. 21st Int. Symp. Fault-Tolerant Computing, June 1991.
6. Stahl M., Buskens R., Bianchini R. On-line diagnosis in general topology networks // IEEE Workshop on Fault-Tolerant Parallel and Distributed Systems, July 1992. – pp. 114–121.
7. Bianchini R., Stahl M., Buskens R. The Adapt2 on-line diagnosis algorithm for general topology networks // Proc. Globecorn, 1992. – pp. 610–614.
8. Jump R.J. YACSIM: Reference Manual, V. 2.1. [Электронный ресурс]. – March, 1993. – Режим доступа: http://oucsace.cs.ohiou.edu/~avinashk/classes/ee663/yac.ps (24.09.2011).