РОССИЙСКАЯ АКАДЕМИЯ НАУК
СИБИРСКОЕ ОТДЕЛЕНИЕ
А В Т О М Е Т Р И Я
№ 2 2001

УДК 681.3.06 : 681.323

М. С. Тарков
(Новосибирск)

САМОСТАБИЛИЗАЦИЯ ПОКРЫВАЮЩЕГО ДЕРЕВА В МАКРОСТРУКТУРЕ РАСПРЕДЕЛЕННОЙ ВЫЧИСЛИТЕЛЬНОЙ СИСТЕМЫ

Предложен протокол, обеспечивающий восстановление системы взаимодействующих процессов, имеющей структуру оптимального корневого дерева, покрывающего граф распределенной вычислительной системы, в котором расстояние от любой вершины дерева до корневой минимально. Данный протокол обеспечивает восстановление оптимального покрывающего дерева как при устойчивых, так и при перемежающихся отказах (сбоях) процессоров и межпроцессорных линий связи вычислительной системы.

Введение. Распределенная вычислительная система (ВС) представляет собой совокупность элементарных машин (ЭМ), объединенных сетью линий связи. Каждая ЭМ является композицией, включающей, как минимум, процессор с памятью и аппаратуру для обеспечения межмашинных взаимодействий. Под макроструктурой распределенной ВС понимается граф, вершинами которого являются ЭМ, а ребрами – линии связи между ними.

Децентрализованные алгоритмы управления ресурсами живучей распределенной вычислительной системы предполагают наличие распределенного по ее элементарным машинам описания макроструктуры ВС, инвариантного по отношению к макроструктуре и допускающего множественные отказы ЭМ и межмашинных линий связи. Таким требованиям удовлетворяет описание структуры взаимосвязей между исправными ЭМ в виде (остовного) дерева, покрывающего макроструктуру ВС, поскольку такое описание можно реализовать на произвольной связной макроструктуре ВС. Покрывающее дерево может быть отображено простым бесперебoрным алгоритмом в произвольный связный граф ВС независимо от числа неисправных вершин и ребер и от их распределения по графу ВС [1–3].

В данной работе предлагается протокол самостабилизации [4–6] корневого покрывающего дерева, обеспечивающий его восстановление как при устойчивых, так и при перемежающихся отказах элементарных машин и межмашинных линий связи. В случае, когда граф ВС распадается на несколько связных компонентов, покрывающее дерево автоматически строится для каждого его компонента.

Термин «самостабилизация» был предложен Дейкстрой [4] для обозначения систем, обладающих свойством достигать требуемого конечного со-

стояния при произвольном начальном. Протоколы самостабилизации привлекательны по двум причинам. Во-первых, они обнаруживают и исправляют локальные ошибки в ЭМ и линиях связи без использования глобальной информации о состоянии системы. Во-вторых, протокол самостабилизации не нуждается в задании определенного начального состояния ВС. Каждая ЭМ может стартовать из произвольного состояния. Это позволяет протоколу самостабилизации исправлять последствия сбоев (перемежающихся отказов).

В работе [5] описан протокол циркуляции фишек (токенов) в однородной анонимной ВС, обеспечивающий самостабилизацию покрывающего дерева при сбоях, искажающих его описание. Под анонимной понимается такая ВС, в которой все вершины изначально не имеют идентификаторов. В связи с этим возникает проблема выделения в графе ВС вершины-лидера, используемой далее в качестве корневой вершины покрывающего дерева. В [5] выделение вершины-лидера основано на том, что число вершин графа является простым. Однако в тех случаях, когда вершины выходят из строя (исключаются из графа) в произвольный момент времени, условие простоты начального числа вершин теряет смысл. Кроме того, в [5] предполагается, что по системе циркулирует только одна фишка. Следовательно, время восстановления покрывающего дерева определяется временем обхода графа ВС и пропорционально числу вершин графа.

В работе [6] предложен децентрализованный алгоритм самостабилизации покрывающего дерева в распределенных ВС с динамически изменяющейся топологией (допускаются отказы и восстановления межмашинных линий связи ВС). В этом алгоритме любая вершина $i \in \{0,1,\ldots,|V|-1\}$ (V – число вершин графа ВС) при обнаружении неправильного состояния инициирует процесс построения покрывающего дерева, помечая его своим идентификатором Id_i. Дерево с большим идентификатором Id_i «захватывает» дерево с меньшим идентификатором $\mathrm{Id}_j < \mathrm{Id}_i$, $i \neq j$, т. е. вершина j дерева с меньшим идентификатором, соседствующая с вершиной дерева с большим идентификатором, присоединяется к ней и получает ее идентификатор Id_i. В конце концов дерево с наибольшим идентификатором «побеждает» другие деревья и покрывает связный компонент графа ВС. В этом алгоритме предполагается, что при попытке присоединения вершины к строящемуся дереву она посылает запрос в его корень и ждет ответа, что, во-первых, приводит к большим временным затратам и, во-вторых, существенно усложняет алгоритм самостабилизации покрывающего дерева. Эти недостатки преодолеваются в протоколе, предлагаемом в данной работе. Упрощение протокола достигается за счет того, что вместо присоединения побежденного дерева к дереву-победителю в целом его вершины присоединяются независимо друг от друга, поэтому необходимость в вышеуказанном запросе отпадает. Время образования покрывающего дерева пропорционально диаметру графа ВС.

Протокол самостабилизации представляет собой набор правил, выполняемых в каждой вершине графа. Каждое правило имеет две части – предикат и действие. Предикат определен на параметрах состояния данной вершины и ее соседей. При выполнении протокола предикаты правил вычисляются в порядке возрастания номеров правил циклически (по достижении максимального номера происходит возврат к минимальному) и бесконечно (протокол должен быть реализован процессом, который постоянно активен). Если предикат какого-либо правила оказывается истинным, то выполняется соответствующее действие, после чего происходит переход к анализу следующе-

го по порядку предиката. В отличие от протокола Дейкстры, предполагающего выполнение правила в каждый момент времени только для одной вершины, здесь предполагается параллельное (одновременное) и асинхронное функционирование вершин. Система предполагается однородной, т. е. все вершины функционируют одинаково (по одним и тем же правилам), но каждая вершина имеет уникальный неизменный идентификатор. Идентификаторы вершин и собственно протокол хранятся в постоянной памяти и поэтому не могут быть искажены.

1. **Определение покрывающего дерева.** Пусть Id_i – уникальный идентификатор вершины i, $\mathrm{Id}_i \neq \mathrm{Id}_j$, $i \neq j$.

Покрывающее ориентированное восходящее к корню дерево ST для каждой вершины $i \in V = \{0,1,\ldots,|V|-1\}$ графа ВС может быть описано тройкой $\langle F_i, R_i, D_i\rangle$, где F_i – идентификатор вершины, которая является «родителем» вершины i; R_i – вершина, являющаяся корнем дерева, которому принадлежит вершина i; если вершина i является корневой, то $F_i = R_i = \mathrm{Id}_i$; D_i – расстояние от данной вершины i до корня дерева.

Пусть $m(\mathrm{Id}) = \min\limits_{i \in V} \mathrm{Id}_i$, $V = \{0,1,\ldots,|V|-1\}$, $|V|$ – число вершин графа ВС.

Для корневой вершины $r \in V$ покрывающего дерева полагаем $\mathrm{Id}_r = m(\mathrm{Id})$, $F_r = \mathrm{Id}_r$, $D_r = 0$. Параметры D_i, F_i дерева удовлетворяют следующим ограничениям:

$$D_i = \begin{cases} 0, & \text{если } \mathrm{Id}_i = m(\mathrm{Id}), \\ D[F_i]+1, & \text{если } \mathrm{Id}_i \neq m(\mathrm{Id}), \end{cases} \tag{1}$$

где $D[F_i]$ – расстояние до корня от вершины, являющейся родительской для вершины i;

$$F_i = \begin{cases} \mathrm{Id}_i, & \text{если } \mathrm{Id}_i = m(\mathrm{Id}), \\ \mathrm{Id}_j, & \text{если } \mathrm{Id}_i \neq m(\mathrm{Id}), \end{cases} \tag{2}$$

где $\mathrm{Id}_j \in Nb_i \subseteq \{0,1,\ldots,\mathrm{v}_i - 1\}$, v_i – степень i-й вершины графа, Nb_i – множество исправных соседей $j \in V$, $j \neq i$, вершины $i \in V$. Согласно свойству (1) граф ST не содержит циклов. Свойство (2) означает, что корневая вершина единственная и имеет идентификатор $\mathrm{Id}_r = m(\mathrm{Id})$, т. е. граф ST, описываемый тройкой $\langle F_i, R_i, D_i\rangle$, не является лесом. Таким образом, граф, обладающий свойствами (1) и (2), является деревом.

С целью создания протокола самостабилизации покрывающего дерева для каждой вершины $i \in V$ вводятся две тройки параметров: $\langle F_i, R_i, D_i\rangle$ – параметры вершины, которые мы будем называть текущими; $\langle F_i^{\lim}, R_i^{\lim}, D_i^{\lim}\rangle$ – параметры вершины, которые мы будем называть предельными, а также вводится список H_i, в который помещается одна тройка параметров $\langle F_{i_k}, R_{i_k}, D_{i_k}\rangle$ для каждого корневого дерева R_{i_k}, в которое попадает вершина i при работе протокола. Список H_i вводится для того, чтобы вершина i могла войти в любое корневое дерево не более одного раза. Именно вершина i получает тройку параметров $\langle F_{i_k}, R_{i_k}, D_{i_k}\rangle$ в качестве текущей, если только в списке H_i нет тройки с параметром R_{i_k} либо есть тройка $\langle F_{i_l}, R_{i_l}, D_{i_l}\rangle$ с параметром $R_{i_l} = R_{i_k}$, но с расстоянием до корня $D_{i_l} \geq D_{i_k}$. При этом «старая» тройка с параметрами $R_{i_l} = R_{i_k}$ и D_{i_l} удаляется из списка H_i, а вместо нее в

список H_i помещается новая тройка с параметрами $R_{i_l} = R_{i_k}$ и D_{i_k}. Это свойство списка H_i использовано при построении приведенного ниже предиката NewFather_{ij}.

Текущие параметры вершины изменяются на каждом цикле работы протокола даже тогда, когда в системе не возникает отказов:

1) если $\text{NewFather}_{ij} = \text{true}$, $j \in Nb_i$, то вершина j имеет минимальные среди всех $k \in Nb_i$ значения R_j и D_j и текущие параметры вершины i получают новые значения по следующему правилу (между соседними вершинами i и j устанавливается связь типа «дочерняя–родительская»):

$$\langle F_i, R_i, D_i\rangle := \langle \text{Id}_j, R_j, D_j + 1\rangle,$$

т. е. выполняется $F_i := \text{Id}_j$, $R_i := R_j$, $D_i := D_j + 1$;

2) если $\text{NewFather}_{ij} = \text{false}$, то текущие параметры получают начальные значения:

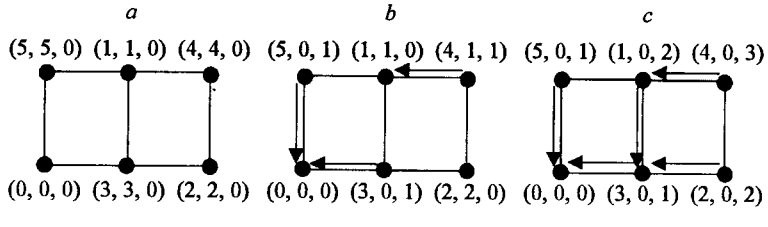
$$\langle F_i, R_i, D_i\rangle := \langle \text{Id}_i, \text{Id}_i, 0\rangle,$$

т. е. выполняется $F_i := \text{Id}_i$, $R_i := \text{Id}_i$, $D_i := 0$.

Предельные параметры (по правилу Rule2) получают значения текущих параметров, если последние совпадают с параметрами одной из троек списка H_i. Предикат NewFather_{ij} задан таким образом, что при отсутствии возмущений вершина i за конечное время приходит в состояние, в котором к моменту выполнения правила Rule2 текущие параметры в вершине i задают покрывающее дерево. Таким образом, при отсутствии возмущений предельные параметры вершины i с некоторого момента времени постоянны и задают покрывающее дерево с корнем в вершине r, причем путь от любой вершины i до вершины r кратчайший. Благодаря периодическому возвращению текущих параметров вершины к начальным значениям любые искажения текущих и предельных параметров исправляются за конечное время.

На рисунке приведен простой пример процесса построения покрывающего дерева при произвольном начальном расположении идентификаторов вершин графа. Каждая вершина помечена параметрами (Id_i, R_i, D_i), $i = 0,1,\ldots$ ..., 5. Параметры F_i представлены дугами. На рисунке *a* показано начальное состояние графа. На рисунке *b* представлено состояние графа после первого цикла выполнения протокола: дерево 0 включает вершины 0, 3 и 5; дерево 1 содержит вершины 1 и 4; дерево 2 содержит только одну вершину 2. На втором цикле протокола (рисунок *c*) вершины деревьев 1 и 2 включаются в дерево 0 (дерево 0 «побеждает» деревья 1 и 2).

2. **Протокол самостабилизации покрывающего дерева.** В нижеописанном протоколе (Rule0, Rule1, Rule2) предполагается использование примитива Par_Exchange для определения наличия или отсутствия исправных вершин $j \in Nb_i$, соседних с данной вершиной $i \in V$, для передачи этим вершинам параметров F_i, R_i, D_i и получения от них параметров F_j, R_j, D_j. Прими-



тив Par_Exchange является блокирующим, т. е. вершина, выполняющая этот примитив, может продолжить выполнение протокола только по завершении обмена параметрами с ее исправными соседями. Это свойство примитива Par_Exchange используется для синхронизации функционирования вершин графа ВС при выполнении протокола самостабилизации. Блокировка не является абсолютной, а ограничена временем тайм-аута. По тайм-ауту соседняя вершина, с которой не удался обмен параметрами, исключается из множества Nb_i.

Пусть $M_i = \arg\min_{k \in Nb_i} R_k$, т. е. M_i – множество вершин $k \in Nb_i$, для которых значение R_k минимально, и пусть $j_m = \arg\min_{k \in M_i} D_k$, т. е. j_m равен такому элементу k множества M_i, для которого минимально значение D_k. Предикат

$$\text{NewFather}_{ij_m} = (R_{j_m} < R_i) \wedge ((\forall i_k \in H_i \mid R_{i_k} \neq R_{j_m}) \vee$$

$$\vee (\exists i_k \in H_i \mid (R_{i_k} = R_{j_m}) \wedge (D_{i_k} \geq D_{j_m} + 1)))$$

принимает значение true, если: 1) корень R_{j_m} дерева вершины j_m меньше корня R_i дерева вершины i; 2) либо для всех троек параметров списка H_i выполнено $R_{i_k} \neq R_{j_m}$, либо в H_i существует тройка параметров i_k, для которой выполнено $R_{i_k} = R_{j_m}$ и $D_{i_k} \geq D_{j_m} + 1$. Таким образом, предикат NewFather_{ij_m} истинен, если возможно уменьшение значения корня R_i до R_{j_m}, причем вершина i либо еще не входила в дерево с корнем R_{j_m}, либо входила, имея параметр $D_i \geq D_{j_m} + 1$.

Для сокращения записи протокола в нем наряду с формой if...then... применена форма записи if...then...else... .

Rule0: if $S_i = 0$
then begin $\langle F_i, R_i, D_i \rangle := \langle \text{Id}_i, \text{Id}_i, 0 \rangle$; Par_Exchange; $S_i := 1$; end

Параметр $S_i \in \{0,1\}$. При $S_i = 0$ параметры $\langle F_i, R_i, D_i \rangle$ вершины $i \in V$ согласно правилу Rule0 получают начальные значения $\langle \text{Id}_i, \text{Id}_i, 0 \rangle$. Далее посредством примитива Par_Exchange параметры $\langle F_i, R_i, D_i \rangle$ передаются исправным соседям вершины $i \in V$. Кроме того, примитив Par_Exchange принимает параметры $\langle F_j, R_j, D_j \rangle$ от исправных соседних вершин $j \in Nb_i$. По завершении Par_Exchange в Rule0 параметр S_i получает значение 1.

Rule1: if $S_i = 1 \wedge \text{NewFather}_{ij_m}$
then begin $\langle F_i, R_i, D_i \rangle := \langle \text{Id}_{j_m}, R_{j_m}, D_{j_m} + 1 \rangle$; Par_Exchange; end
else $S_i := 0$;

Если при $S_i = 1$ выполнено NewFather_{ij_m} = true, то текущие параметры вершины $i \in V$ изменяются: F_i становится равным идентификатору Id_{j_m} вершины j_m, R_i становится равным корню R_{j_m} дерева, которому принадлежит вершина j_m, расстояние D_i до корня R_{j_m} получает значение $D_{j_m} + 1$. После

изменения значений параметров вершина $i \in V$ выполняет примитив Par_Exchange. Если же $\mathrm{NewFather}_{ij_m}$ = false, то S_i получает значение 0.

Rule2: if $\langle F_i, R_i, D_i\rangle = \langle F_i, R_i, D_i\rangle\big|_{H_i}$

then begin $\langle F_i^{\lim}, R_i^{\lim}, D_i^{\lim}\rangle := \langle F_i, R_i, D_i\rangle$; Tree := 1 end

else begin delete $(H_i, \langle \_, R_i, \_\rangle)$; insert $(H_i, \langle F_i, R_i, D_i\rangle)$; Tree := 0; end

Если текущие параметры вершины $i \in V$ совпадают с параметрами одной из троек списка H_i, то предельным параметрам вершины i присваиваются соответствующие значения текущих параметров и двоичная переменная Tree получает значение 1. Иначе из списка H_i примитивом delete удаляется тройка, у которой значение корня равно R_i, и далее в список H_i помещается тройка текущих параметров $\langle F_i, R_i, D_i\rangle$. При этом переменная Tree получает значение 0. Нетрудно видеть, что предельные параметры вершины получают значения текущих, если последние совпадают перед выполнением правила Rule2 для двух непосредственно следующих друг за другом циклов протокола.

3. **Корректность протокола.** Введем обозначения: l_{ij} – расстояние между вершинами i и j, $l_{ij} = l_{ji}$; $L_r = \max\limits_{i \in V} l_{ri}$ – расстояние от вершины r до наиболее удаленной от нее вершины; V_{rl} – множество вершин, удаленных от вершины r на расстояние l (будем называть V_{rl} l-м ярусом графа ВС относительно вершины r); $V = \bigcup\limits_{l=0}^{L_r} V_{rl}$, $V_{r0} = \{r\}$, $V_{r1} = Nb_r$.

Определение. Множество $\{\langle\langle F_i, R_i, D_i\rangle, \langle F_i^{\lim}, R_i^{\lim}, D_i^{\lim}\rangle\rangle \mid i \in V\}$ будем называть состоянием ВС.

Далее в качестве состояния ВС на каждом шаге (цикле) протокола будем рассматривать состояние, предшествующее выполнению правила Rule1, так как в этот момент каждая вершина имеет информацию о своих соседях, необходимую для принятия решения об изменении своего состояния.

Определение. Состояние ВС будем называть устойчивым, если:

1) вершина r, для которой

$$\mathrm{Id}_r = \min_{i \in V} \mathrm{Id}_i, \tag{3}$$

имеет состояние

$$\mathrm{State}_r = \langle\langle F_r, R_r, D_r\rangle, \langle F_r^{\lim}, R_r^{\lim}, D_r^{\lim}\rangle\rangle = \langle\langle \mathrm{Id}_r, \mathrm{Id}_r, 0\rangle, \langle \mathrm{Id}_r, \mathrm{Id}_r, 0\rangle\rangle; \tag{4}$$

2) любая пара вершин (i, j), $i \neq j$, такая, что

$$(i, j) \in E, G = (V, E), \quad i \in V_{rl},\ l = 1, 2, \ldots, L_r,\ j \in V_{r,l-1}, \tag{5}$$

находится в одном из следующих состояний:

$$\langle\langle F_i, R_i, D_i\rangle, \langle F_i^{\lim}, R_i^{\lim}, D_i^{\lim}\rangle\rangle = \langle\langle \mathrm{Id}_j, \mathrm{Id}_r, D_j + 1\rangle, \langle \mathrm{Id}_j, \mathrm{Id}_r, D_j + 1\rangle\rangle,$$

$\mathrm{State}_{ij}(1)$:

$$\langle\langle F_j, R_j, D_j\rangle, \langle F_j^{\lim}, R_j^{\lim}, D_j^{\lim}\rangle\rangle = \langle\langle \mathrm{Id}_j, \mathrm{Id}_j, 0\rangle, \langle \mathrm{Id}_k, \mathrm{Id}_r, D_j\rangle\rangle; \tag{6}$$

$$\langle\langle F_i, R_i, D_i\rangle, \langle F_i^{\lim}, R_i^{\lim}, D_i^{\lim}\rangle\rangle = \langle\langle \mathrm{Id}_i, \mathrm{Id}_i, 0\rangle, \langle \mathrm{Id}_j, \mathrm{Id}_r, D_j + 1\rangle\rangle,$$

$\mathrm{State}_{ij}(2)$:

$$\langle\langle F_j, R_j, D_j\rangle, \langle F_j^{\lim}, R_j^{\lim}, D_j^{\lim}\rangle\rangle = \langle\langle \mathrm{Id}_k, \mathrm{Id}_r, D_j\rangle, \langle \mathrm{Id}_k, \mathrm{Id}_r, D_j\rangle\rangle, \quad (7)$$

где $k = r$ при $l = 1,2$ или $k \in V_{r,l-2}$ при $l > 2$, а D_j – длина кратчайшего пути от вершины $j \neq r$ до вершины r.

Утверждение 1. Протокол не выводит систему из устойчивого состояния, причем на каждом цикле выполнения протокола произвольная пара вершин (i, j), удовлетворяющая (5), переходит из состояния $\mathrm{State}_{ij}(1)$ в состояние $\mathrm{State}_{ij}(2)$ или обратно, а состояние вершины r остается неизменным.

Доказательство.

1. Из (3) и (4) следует $\mathrm{NewFather}_{rj} = \mathrm{false}$ для всех $i \in V_{rl}$. Следовательно, правило Rule1 не изменит текущее состояние вершины r. Тогда правило Rule2 не изменит предельное состояние вершины r, т. е. в любой момент времени справедливо (4).

Рассмотрим произвольную вершину $i \in V_{r1}$. Согласно условиям (6), (7) утверждения 1 она изначально (при $t = t_0$) находится либо в состоянии

$$\mathrm{State}_i(1): \langle\langle F_i, R_i, D_i\rangle, \langle F_i^{\lim}, R_i^{\lim}, D_i^{\lim}\rangle\rangle = \langle\langle \mathrm{Id}_r, \mathrm{Id}_r, 1\rangle, \langle \mathrm{Id}_r, \mathrm{Id}_r, 1\rangle\rangle, \quad (8)$$

либо в состоянии

$$\mathrm{State}_i(2): \langle\langle F_i, R_i, D_i\rangle, \langle F_i^{\lim}, R_i^{\lim}, D_i^{\lim}\rangle\rangle = \langle\langle \mathrm{Id}_i, \mathrm{Id}_i, 0\rangle, \langle \mathrm{Id}_r, \mathrm{Id}_r, 1\rangle\rangle. \quad (9)$$

Из (3) и (8) следует, что если вершина $i \in V_{r1}$ находится в состоянии $\mathrm{State}_i(1)$, то минимизация $R_i = \mathrm{Id}_r$ невозможна ($\mathrm{NewFather}_{ij_m} = \mathrm{false}$), поэтому согласно Rule1 параметр S_i получает значение 0, правило Rule2 не меняет состояние $\mathrm{State}_i(1)$ и правило Rule0 переводит текущие параметры вершины i в состояние $\langle \mathrm{Id}_i, \mathrm{Id}_i, 0\rangle$, т. е. вершина i переходит в состояние $\mathrm{State}_i(2)$.

Из (3) и (9) следует, что если вершина $i \in V_{r1}$ находится в состоянии $\mathrm{State}_i(2)$, то $\mathrm{NewFather}_{ir} = \mathrm{true}$. Согласно Rule1 параметр S_i остается равным 1, и правило Rule1 переводит текущие параметры вершины i в состояние $\langle \mathrm{Id}_r, \mathrm{Id}_r, 1\rangle$, т. е. вершина i переходит в состояние $\mathrm{State}_i(1)$, которое правилом Rule2 не изменяется.

2. Пусть утверждение 1 справедливо для произвольной пары вершин (i, j), удовлетворяющей (5) при некотором значении $l \in 1, \ldots, L_r - 1$. Докажем его справедливость для пары вершин (u, i), удовлетворяющей (5) при $l + 1$, т. е. для

$$(u, i) \in E, G = (V, E), \quad i \in V_{rl}, \quad l = 1,2,\ldots,L_r, \quad u \in V_{r,l+1}. \quad (10)$$

Согласно (6) и (7) пара (u, i) изначально находится либо в состоянии

$$\langle\langle F_u, R_u, D_u\rangle, \langle F_u^{\lim}, R_u^{\lim}, D_u^{\lim}\rangle\rangle = \langle\langle \mathrm{Id}_i, \mathrm{Id}_r, D_i + 1\rangle, \langle \mathrm{Id}_i, \mathrm{Id}_r, D_i + 1\rangle\rangle,$$

$\mathrm{State}_{ui}(1)$:

$$\langle\langle F_i, R_i, D_i\rangle, \langle F_i^{\lim}, R_i^{\lim}, D_i^{\lim}\rangle\rangle = \langle\langle \mathrm{Id}_i, \mathrm{Id}_i, 0\rangle, \langle \mathrm{Id}_j, \mathrm{Id}_r, D_i\rangle\rangle, \quad (11)$$

либо в состоянии

$$\langle\langle F_u, R_u, D_u\rangle, \langle F_u^{\lim}, R_u^{\lim}, D_u^{\lim}\rangle\rangle = \langle\langle \mathrm{Id}_u, \mathrm{Id}_u, 0\rangle, \langle \mathrm{Id}_i, \mathrm{Id}_r, D_i+1\rangle\rangle,$$

$\mathrm{State}_{ui}(2)$:

$$\langle\langle F_i, R_i, D_i\rangle, \langle F_i^{\lim}, R_i^{\lim}, D_i^{\lim}\rangle\rangle = \langle\langle \mathrm{Id}_j, \mathrm{Id}_r, D_i\rangle, \langle \mathrm{Id}_j, \mathrm{Id}_r, D_i\rangle\rangle, \quad (12)$$

где $j \in V_{r,l-1} \cap Nb_i$.

Если пара вершин (u, i) находится в состоянии $\mathrm{State}_{ui}(1)$, то $R_u = \mathrm{Id}_r$ и в силу (3) не может уменьшаться ($\mathrm{NewFather}_{uj} = \mathrm{false}$ для всех $j \in Nb_u = V_{ul}$). Легко видеть, что в результате последовательного срабатывания правил Rule1, Rule2 и Rule0 вершина u переходит в состояние

$$\mathrm{State}_u(2): \langle\langle F_u, R_u, D_u\rangle, \langle F_u^{\lim}, R_u^{\lim}, D_u^{\lim}\rangle\rangle = \langle\langle \mathrm{Id}_u, \mathrm{Id}_u, 0\rangle, \langle \mathrm{Id}_i, \mathrm{Id}_r, D_i+1\rangle\rangle.$$

Пусть пара вершин (u, i) находится в состоянии $\mathrm{State}_{ui}(2)$. Согласно (3) параметр $R_i = \mathrm{Id}_r$ является минимальным из всех возможных, т. е. $i \in M_u \subseteq \subseteq Nb_u$, и по условию утверждения 1 величина D_i минимальна. Следовательно, на паре вершин (u, i) предикат $\mathrm{NewFather}_{ui} = \mathrm{true}$, правило Rule1 сделает текущее состояние вершины u равным $\langle \mathrm{Id}_i, \mathrm{Id}_r, D_i + 1\rangle$ и после выполнения Rule2 вершина u перейдет в состояние

$$\mathrm{State}_u(1): \langle\langle F_u, R_u, D_u\rangle, \langle F_u^{\lim}, R_u^{\lim}, D_u^{\lim}\rangle\rangle = \langle\langle \mathrm{Id}_i, \mathrm{Id}_r, D_i+1\rangle, \langle \mathrm{Id}_i, \mathrm{Id}_r, D_i+1\rangle\rangle.$$

Утверждение 1 доказано.

Следствие. Если ВС находится в устойчивом состоянии, то предельные значения параметров вершин не изменяются протоколом и задают покрывающее дерево.

Утверждение 2. При произвольных начальных значениях параметров вершин протокол приводит ВС в устойчивое состояние за конечное число циклов.

Доказательство.

1. Пусть параметры вершин ВС S_i, $\langle F_i, R_i, D_i\rangle$, $\langle F_i^{\lim}, R_i^{\lim}, D_i^{\lim}\rangle$, H_i имеют случайные значения (например, вследствие искажений) в начальный момент времени t_0. Покажем, что существует конечный момент времени $t_1 > t_0$, когда для всех вершин $i \in V$ выполнено $F_i, R_i \in \{\mathrm{Id}_k \mid k = 1, \ldots, |V|\}$; $D_i \le L_r$. Рассмотрим произвольную вершину $i \in V$, для которой $R_i \notin \{\mathrm{Id}_k \mid k = 1, \ldots, |V|\}$.

Если $S_i = 0$, то согласно Rule0 получаем $\langle F_i, R_i, D_i\rangle := \langle \mathrm{Id}_i, \mathrm{Id}_i, 0\rangle$, иначе (т. е. если $S_i = 1$, так как S_i – однобитовая переменная) Rule0 не срабатывает. Далее вычисляется предикат $\mathrm{NewFather}_{ij_m}$. Если $\mathrm{NewFather}_{ij_m} = \mathrm{false}$, то в результате выполнения Rule1, Rule2 и Rule0 имеем

$$\langle F_i, R_i, D_i\rangle\big|_{t = t_0 + 1} = \langle \mathrm{Id}_i, \mathrm{Id}_i, 0\rangle.$$

Если же $\mathrm{NewFather}_{ij_m} = \mathrm{true}$, то параметры R_i, D_i могут оставаться искаженными, так как согласно Rule1 новые значения R_i, D_i вычисляются через параметры R_{j_m}, D_{j_m} соседней вершины j_m:

$$\langle F_i, R_i, D_i\rangle := \langle \mathrm{Id}_{j_m}, R_{j_m}, D_{j_m} + 1\rangle.$$

Рассмотрим множество параметров $\{R_i \mid i = 1, \ldots, |V|\}$. Пусть

$$R_s = \min_i R_i < \mathrm{Id}_r. \tag{13}$$

Для вершины s, удовлетворяющей (13), всегда $\mathrm{NewFather}_{sj_m} = \mathrm{false}$ и, следовательно, в результате выполнения цикла протокола получаем

$$\langle F_s, R_s, D_s \rangle\big|_{t = t_0 + 1} = \langle \mathrm{Id}_s, \mathrm{Id}_s, 0 \rangle. \tag{14}$$

Рассмотрим вершину $i \in V_{s1} = Nb_s$. На первом цикле выполнения протокола при $S_i = 0$ и $\mathrm{NewFather}_{is} = \mathrm{false}$ $(s = j_m \in Nb_i)$ получаем

$$\langle F_i, R_i, D_i \rangle\big|_{t = t_0 + 1} = \langle \mathrm{Id}_i, \mathrm{Id}_i, 0 \rangle. \tag{15}$$

Иначе (при $\mathrm{NewFather}_{is} = \mathrm{true}$) имеем

$$\langle F_i, R_i, D_i \rangle\big|_{t = t_0 + 1} = \langle \mathrm{Id}_s, R_s, D_s + 1 \rangle, \tag{16}$$

т. е. вершина i получила искаженные значения параметров $R_i = R_s$ и $D_i = D_s + 1$. Тройка параметров $\langle F_i, R_i, D_i \rangle = \langle \mathrm{Id}_s, R_s, D_s + 1 \rangle$ заносится в список H_i (при этом «старая» тройка с параметром R_s из списка H_i удаляется). В силу (13) $R_i = R_s$ не может далее уменьшаться, следовательно, после выполнения (16) имеем $\mathrm{NewFather}_{is} = \mathrm{false}$, что приводит к

$$\langle F_i, R_i, D_i \rangle\big|_{t = t_0 + 2} = \langle \mathrm{Id}_i, \mathrm{Id}_i, 0 \rangle.$$

Итак, в вершине s на шаге $t = t_0 + 1$ и в любой вершине $i \in V_{s1} = Nb_s$ не позднее, чем на шаге $t = t_0 + 2$, искажения текущих параметров будут ликвидированы.

Следует отметить, что вершина $i \in V_{s1} = Nb_s$ при $t > t_0 + 2$ не может вторично получить значение $R_i = R_s$, так как согласно (14) в вершине s искажения ликвидированы на шаге $t = t_0 + 1$, а в любой вершине $j \neq s$, $j \in Nb_i$, согласно правилу Rule1 тройка, содержащая $R_j = R_s$, будет иметь $D_j \geq D_s + 1$, что приведет к выполнению $\mathrm{NewFather}_{ij} = \mathrm{false}$.

2. Пусть в момент времени $t = t_0 + l$, $l \in \{1, \ldots, L_s - 1\}$, вершина $i \in V_{sl}$ имеет параметры

$$\langle F_i, R_i, D_i \rangle\big|_{t = t_0 + l} = \langle \mathrm{Id}_j, R_s, D_s + l \rangle, \tag{17}$$

где $j \in V_{s,l-1} \cap Nb_i$ (при $l = 1$ параметры (17) совпадают с (16)). В силу (13) $\mathrm{NewFather}_{ik}\big|_{t = t_0 + l} = \mathrm{false}$ $(k \in Nb_i)$, поэтому

$$\langle F_i, R_i, D_i \rangle\big|_{t = t_0 + l + 1} = \langle \mathrm{Id}_i, \mathrm{Id}_i, 0 \rangle.$$

Рассмотрим произвольную вершину $j \in Nb_i \cap V_{s,l+1}$ в момент времени $t = t_0 + l$. Согласно протоколу значения параметров из вершины s за время l (l циклов протокола) могут быть переданы на расстояние не большее, чем l. Поэтому в момент времени $t = t_0 + l$ для вершины $j \in Nb_i \cap V_{s,l+1}$ в силу (13) могут выполняться $R_j > R_i = R_s$ и, возможно, $\mathrm{NewFather}_{ji} = \mathrm{true}$ (наихудший случай). Тогда из (17) следует

$$\langle F_j, R_j, D_j \rangle\big|_{t = t_0 + l + 1} = \langle \mathrm{Id}_i, R_s, D_s + l + 1 \rangle. \tag{18}$$

В силу (13) $R_j = R_s$ не может далее уменьшаться, и, следовательно, после выполнения (18) имеем $\mathrm{NewFather}_{is} = \mathrm{false}$, что приводит к

$$\langle F_j, R_j, D_j\rangle\big|_{t=t_0+l+2} = \langle \mathrm{Id}_j, \mathrm{Id}_j, 0\rangle. \qquad (19)$$

Аналогично п. 1 доказательства легко убедиться в том, что вершина $j \in Nb_i \cap V_{s,l+1}$ не может вторично получить значение $R_j = R_s\big|_{t=t_0}$, так как передача значения R_s от любой вершины к соседней сопровождается увеличением параметра расстояния до вершины s, следовательно, для любой вершины $i \in Nb_j$, имеющей $R_i = R_s\big|_{t>t_0+l+2}$, будет выполнено $D_i\big|_{t>t_0+l+2} > D_j\big|_{t=t_0+l+2}$, что неизбежно приведет к выполнению $\mathrm{NewFather}_{ji}\big|_{t>t_0+l+2} = \mathrm{false}$.

Из произвольности $l \in \{1, \ldots, L_s - 1\}$ следует, что не позднее момента времени $t_1 = t_0 + L_s + 2$ текущие параметры любой вершины $i \in V$ графа ВС удовлетворяют ограничениям

$$F_i \in \{\mathrm{Id}_k \mid k = 1, \ldots, |V|\}, \quad R_i \in \{\mathrm{Id}_k \mid k = 1, \ldots, |V|\}, \quad D_i \le L_s. \qquad (20)$$

3. Из (3), (19), (20) вытекает

$$\langle F_r, R_r, D_r\rangle\big|_{t>t_1} = \langle \mathrm{Id}_r, \mathrm{Id}_r, 0\rangle. \qquad (21)$$

Рассмотрим вершину $i \in Nb_r = V_{r1}$. Если при $t = t_1$ выполнено $\mathrm{NewFather}_{ij_m} = \mathrm{false}$, то при $t = t_1 + 1$ выполнено

$$\langle F_i, R_i, D_i\rangle\big|_{t=t_1+1} = \langle \mathrm{Id}_j, \mathrm{Id}_i, 0\rangle, \qquad (22)$$

и в силу (20)–(22) имеем $\mathrm{NewFather}_{ij_m}\big|_{t=t_1+2} = \mathrm{NewFather}_{ir} = \mathrm{true}$ и

$$\langle F_i, R_i, D_i\rangle\big|_{t=t_1+2} = \langle \mathrm{Id}_r, \mathrm{Id}_r, 1\rangle.$$

Иначе $\mathrm{NewFather}_{ij_m}\big|_{t=t_1+1} = \mathrm{NewFather}_{ir} = \mathrm{true}$ и

$$\langle F_i, R_i, D_i\rangle\big|_{t=t_1+1} = \langle \mathrm{Id}_r, \mathrm{Id}_r, 1\rangle. \qquad (23)$$

Итак, при $t = t_1 + 1$ для $i \in Nb_r = V_{r1}$ имеет место либо (22), либо (23). Продолжая рассуждения, нетрудно убедиться в том, что (22) или (23) выполняется для любых $t \ge t_1 + 1$. Кроме того, из (21)–(23) и правила Rule2 следует, что

$$\langle F_i^{\lim}, R_i^{\lim}, D_i^{\lim}\rangle\big|_{t>t_1+2} = \langle \mathrm{Id}_r, \mathrm{Id}_r, 1\rangle.$$

Таким образом, при $t > t_1 + 2$ вершина $i \in Nb_r = V_{r1}$ находится либо в состоянии

$$\mathrm{State}_i(1): \langle\langle F_i, R_i, D_i\rangle, \langle F_i^{\lim}, R_i^{\lim}, D_i^{\lim}\rangle\rangle = \langle\langle \mathrm{Id}_r, \mathrm{Id}_r, 1\rangle, \langle \mathrm{Id}_r, \mathrm{Id}_r, 1\rangle\rangle,$$

либо в состоянии

$$\text{State}_i(2)\colon \langle\langle F_i, R_i, D_i\rangle, \langle F_i^{\lim}, R_i^{\lim}, D_i^{\lim}\rangle\rangle = \langle\langle \text{Id}_i, \text{Id}_i, 0\rangle, \langle \text{Id}_r, \text{Id}_r, 1\rangle\rangle,$$

причем на каждом цикле протокола происходит переход из $\text{State}_i(1)$ в $\text{State}_i(2)$ либо наоборот.

4. Пусть в моменты времени $t = t_2 + 2m$, $t_2 = t_1 + l + 2$, $m = 0,1,2,\ldots$, вершина $i \in V_{rl}$ имеет состояние

$$\text{State}_i(1)\colon \langle\langle F_i, R_i, D_i\rangle, \langle F_i^{\lim}, R_i^{\lim}, D_i^{\lim}\rangle\rangle = \langle\langle \text{Id}_k, \text{Id}_r, l\rangle, \langle \text{Id}_k, \text{Id}_r, l\rangle\rangle, \quad (24)$$

а в моменты $t = t_2 + 2m + 1$ – состояние

$$\text{State}_i(2)\colon \langle\langle F_i, R_i, D_i\rangle, \langle F_i^{\lim}, R_i^{\lim}, D_i^{\lim}\rangle\rangle = \langle\langle \text{Id}_i, \text{Id}_i, 0\rangle, \langle \text{Id}_k, \text{Id}_r, 1\rangle\rangle, \quad (25)$$

$k \in V_{r,l-1} \cap Nb_i$, причем на каждом цикле протокола происходит переход из $\text{State}_i(1)$ в $\text{State}_i(2)$ либо наоборот (в п. 3 это показано для $l = 1$).

Рассмотрим произвольную вершину $j \in Nb_i \cap V_{s,l+1}$ в момент времени t_2. Согласно протоколу значения параметров из вершины r за время l могут быть переданы на расстояние не большее, чем l. Поэтому в момент времени t_2 для вершины $j \in Nb_i \cap V_{s,l+1}$ в силу (20) выполняются $R_j > R_i = \text{Id}_r$ и $\text{NewFather}_{ji} = \text{true}$. Тогда из (24) следует

$$\langle F_j, R_j, D_j\rangle\big|_{t=t_2+1} = \langle \text{Id}_i, \text{Id}_r, l+1\rangle. \quad (26)$$

В силу (3) $R_j = \text{Id}_r$ не может далее уменьшаться и, следовательно, после выполнения (26) имеем $\text{NewFather}_{is} = \text{false}$, что приводит к

$$\langle F_j, R_j, D_j\rangle\big|_{t=t_2+2} = \langle \text{Id}_j, \text{Id}_j, 0\rangle. \quad (27)$$

Продолжая рассуждения, убеждаемся в том, что (26) и (27) выполняются соответственно для любых $t = t_2 + 2m + 1$ и $t = t_2 + 2m + 2$. Кроме того, из (24)–(27) и правила Rule2 следует, что при $t \geq t_2 + 2$ выполняется

$$\langle F_j^{\lim}, R_j^{\lim}, D_j^{\lim}\rangle = \langle \text{Id}_i, \text{Id}_r, l+1\rangle.$$

Таким образом, вершина $j \in Nb_i \cap V_{s,l+1}$ находится при $t = t_2 + 2m + 1$, $m = 1,2,\ldots$, в состоянии

$$\text{State}_j(1)\colon \langle\langle F_j, R_j, D_j\rangle, \langle F_j^{\lim}, R_j^{\lim}, D_j^{\lim}\rangle\rangle = \langle\langle \text{Id}_i, \text{Id}_r, l+1\rangle, \langle \text{Id}_i, \text{Id}_r, l+1\rangle\rangle \quad (28)$$

и при $t = t_2 + 2m + 2$, $m = 1,2,\ldots,$ – в состоянии

$$\text{State}_j(2)\colon \langle\langle F_j, R_j, D_j\rangle, \langle F_j^{\lim}, R_j^{\lim}, D_j^{\lim}\rangle\rangle = \langle\langle \text{Id}_j, \text{Id}_j, 0\rangle, \langle \text{Id}_i, \text{Id}_r, l+1\rangle\rangle, \quad (29)$$

причем на каждом цикле протокола происходит переход из $\text{State}_j(1)$ в $\text{State}_j(2)$ либо наоборот. Из (24), (25), (28), (29) следует, что состояние пары вершин (j, i) является устойчивым.

В силу произвольности выбора l утверждение 2 доказано.

4. **Реализация протокола самостабилизации покрывающего дерева.** Протокол самостабилизации покрывающего (остовного) дерева может быть

реализован в каждой ЭМ распределенной ВС в виде постоянно действующего системного процесса repair_tree. Этот процесс будет восстанавливать остовное дерево при условии, что все идентификаторы вершин различны, т. е. $\mathrm{Id}_i \neq \mathrm{Id}_j$, $i \neq j$, $i, j = 0,1,\ldots,|V|-1$. Искажение и, как следствие, совпадение идентификаторов исключаются при их хранении в постоянной памяти. Для извещения о завершении восстановления остовного дерева в процессе repair_tree осуществляется установка семафора Tree = 1. При обнаружении Tree = 0 процессы, использующие покрывающее дерево, блокируются.

В качестве примера использования покрывающего дерева рассмотрим процедуру Sinchro синхронизации системы процессов параллельной программы. Эта процедура используется в тех случаях, когда выполнение параллельной программы состоит из нескольких этапов и переход от одного этапа к следующему за ним возможен лишь тогда, когда все ветви параллельной программы завершили выполнение данного этапа. Операция P(Tree) блокирует процесс, выполняющий процедуру Sinchro в ожидании завершения построения покрывающего дерева. Процедура Upward в каждой вершине дерева $i \in V$ принимает сообщение mes от дочерних вершин посредством оператора Receive и посылает сообщение родительской вершине F_i посредством оператора Send. Соседняя вершина j считается дочерней для вершины i, если значение ее указателя $F_j = \mathrm{Id}_i$. Когда сообщение mes достигает корневой вершины дерева, можно считать, что все ЭМ системы завершили данный этап вычислений и готовы перейти к новому этапу. Процедура Downward дает машинам системы разрешение на переход. Она в каждой вершине дерева принимает сообщение mes от родительской вершины F_i и посылает сообщение дочерним вершинам. Операция передачи Send(Id_j, mes) посылает сообщение mes соседней вершине $j \in Nb_i$. Операция приема Receive(Id_j, mes) соответственно принимает сообщение mes от соседней вершины $j \in Nb_i$.

```
    procedure Sinchro;
    begin P(Tree); Upward; Downward; V(Tree); end
    procedure Upward;
begin for j ∈ Nb_i do if (Id_i = F_j) Receive(Id_j, mes);
    if (F_i ≠ Id_i) Send(F_i, mes); end
      procedure Downward;
      begin if (F_i ≠ Id_i) Receive(F_i, mes);
      for j ∈ Nb_i do if (Id_i = F_j) Send(Id_j, mes); end
```

В случае аварийного завершения процедур приема–передачи (например, по тайм-ауту) происходит передача управления на начало процедуры Sinchro (на операцию P(Tree)). В результате процесс вычислений блокируется в ожидании восстановления покрывающего дерева.

Заключение. Рассмотрены известные подходы к построению протоколов, обеспечивающих восстановление (самостабилизацию) корневого дерева, покрывающего макроструктуру распределенной ВС. Предложен новый протокол самостабилизации покрывающего дерева, учитывающий динамически происходящие отказы и восстановления машин и межмашинных линий связи для ВС, в которых каждая ЭМ обладает уникальным идентификатором. Согласно этому протоколу новые вершины присоединяются к формируемому дереву без посылки запроса в корневую вершину этого дерева. В результате протокол существенно упрощается. Использование протокола самостабилизации дерева для восстановления параллельных вычислений про-

демонстрировано на примере процедуры синхронизации системы параллельных процессов. Данный протокол обеспечивает восстановление покрывающего корневого дерева как при устойчивых, так и при перемежающихся отказах (сбоях) элементарных машин и межмашинных линий связи ВС, причем в этом дереве путь от произвольной вершины до корневой является кратчайшим.

СПИСОК ЛИТЕРАТУРЫ

1. **Тарков М. С.** Параллельная отказоустойчивая обработка изображений в транспьютерной системе МИКРОС-Т // Научное приборостроение. 1995. **5**, № 3–4. С. 74.
2. **Tarkov M. S.** Mapping parallel programs onto distributed robust computer systems // Proc. of the 15th IMACS World Congress on Scientific Computation, Modelling and Applied Mathematics. Berlin, 1997. V. 6: Applications in Modelling and Simulation /Ed. A. Sydow. P. 365.
3. **Тарков М. С.** Параллельный алгоритм бисекции структуры распределенной вычислительной системы // Тр. VI Междунар. семинара «Распределенная обработка информации». Новосибирск, 1998. С. 96.
4. **Dijkstra E. W.** Self-stabilizing systems in spite of distributed control // Commun. ACM. 1974. **17(11)**. P. 643.
5. **Hwang S. T., Wuu L. C.** Self-stabilizing token circulation in uniform networks // Distributed Computing. 1997. **10**, N 4. P. 181.
6. **Afek Y., Kutten S., Yung M.** The local detection paradigm and its applications to self stabilization // Theor. Comput. Sci. 1997. **186**. P. 199.

Институт физики полупроводников СО РАН,
E-mail: tarkov@isp.nsc.ru

Поступила в редакцию
11 мая 1999 г.