© 2012 г. Н.А. ИСАЕВА, к.т.н., с.н.с.,
М.Л. МИЛКОВ, старший математик
(Институт проблем управления
им. В.А. Трапезникова РАН, Москва)

ОРГАНИЗАЦИЯ ДИАГНОСТИРОВАНИЯ ОШИБОК ДЛЯ ОДНОГО ИЗ ВАРИАНТОВ МНОГОВЕРСИОННОГО РЕЗЕРВИРОВАНИЯ ВЗАИМОСВЯЗАННЫХ ПРОГРАММНЫХ МОДУЛЕЙ В УПРАВЛЯЮЩИХ ПАРАЛЛЕЛЬНЫХ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМАХ

Впервые формально обоснована, синтезирована и логически описана структура специализированного (для одного из вариантов многоверсионного резервирования) программного блока диагностирования для определения координаты одиночной ошибки вычислительных ресурсов системы, типа ошибки (сбой или отказ процессора), а также для идентификации программного модуля с искаженными результатами его выполнения.

THE ORGANIZATION OF ERRORS DIAGNOSIS FOR ONE OF MULTIVERSION REDUNDANCY VARIANTS OF INTERDEPENDENT PROGRAM MODULES IN PARALLEL CONTROL COMPUTING SYSTEMS / N.A. Isaeva, M.L. Milkov (V.A. Trapeznikov Institute of Control Sciences, Profsoyuznaya 65, Moscow 117342, Russia, E-mail: nat-i@ipu.ru). Structure specialized (for one of multiversion redundancy variants) program diagnosis unit to determine the coordinates of a single error of system's computing resources, the type of error (failure or fault of the processor), as well as to identify the program module with distorted results of its execution is formally proved, logically synthesized and described for the first time.

1. Введение

В рамках проблемы организации и прогнозирования надежного выполнения комплексов резервированных программных модулей (комплексов взаимосвязанных работ – КВР) со случайными временами их выполнения в управляющих параллельных вычислительных системах [1-3], – в докладе предлагается способ организации диагностирования ошибок для одного из вариантов многоверсионного резервирования (МР) с асинхронным запуском программных модулей КВР.

Рассматривается и развивается один из аспектов асинхронного резервирования процессов – принцип многоверсионного (многовариантного) программирования [4,5]. Данный подход базируется на параллельном асинхронном выполнении неидентичных программ одного и того же процесса.

Многоверсионное программирование представляет собой независимое создание и использование двух или более функционально эквивалентных программ, называемых вариантами (версиями, альтернативами) и соответствующих одной и той же исходной спецификации [6]. Функционально эквивалентные программы – это такие программы, которые по одинаковым допустимым входным данным вырабатывают одинаковые (в общем случае) результаты вычислений. Для различных вариантов решения одной и той же задачи (и/или фрагмента) могут использоваться различные алгоритмы, языки программирования; этим и предполагаются различия во времени выполнения разных версий задачи, а следовательно - асинхронное их выполнение даже на однотипных параллельных процессорах.

Основной областью применения многоверсионного программирования являются программные системы, многократно используемые на параллельных ВС в течение длительного времени, – в особенности такие, к которым предъявляются предельно высокие требования по надежности функционирования, в первую очередь – системы обработки информации и управления в реальном масштабе времени.

Таким образом, механизм многоверсионного программирования может использоваться для достижения, по меньшей мере, двух целей:

– с одной стороны, такой механизм является средством резервирования процессов в параллельных ВС, а, следовательно, – средством обеспечения отказоустойчивости параллельных ВС, а также обеспечения достоверности решения задач;

– с другой стороны, возможность варьирования различными копиями процессов, в зависимости от текущего состояния процессов и ВС в целом, а также от заданного директивного времени (или его “остатка”), способствует возможности организации адаптивного управления процессами [7, 8].

Напомним, что само понятие «многоверсионное резервирование взаимосвязанных программных модулей» проистекает из принципов многоверсионного (многовариантного) резервирования программ, описанных выше, было предложено в ИПУ РАН сравнительно недавно, и потому особенности этого варианта резервирования, его свойства, потенциальные возможности и эффекты (как положительные, так и отрицательные) мало исследованы.

Наиболее распространенными вариантами многоверсионного резервирования комплекса взаимосвязанных программных модулей считаются следующие [9]:

1) Работы-версии реализуют те же алгоритмы, что и исходные работы-оригиналы, и связаны между собою такими же логическими и информационными связями, что и работы исходного КВР (т.е. в целом структура резервного КВР идентична структуре исходного КВР), но при этом в работах-версиях используются другие методы вычислений. Например, каждая работа-версия является вычислительным аналогом исходной работы, но с более низкой точностью расчетов и, следовательно, с меньшим временем выполнения. Примером таких работ может служить случай, когда работа-оригинал использует для вычислений переменные с плавающей точкой, а работа-версия – целочисленные переменные;
2) В резервном КВР полностью сохраняется структура связей исходного КВР, но одноименные программные модули реализуют принципиально различные алгоритмы. Например, каждая работа-версия решает ту же задачу, что и работа-оригинал, но использует абсолютно другой алгоритм её решения. Допустим, что задачей некоего программного модуля КВР является решение системы линейных уравнений. Как известно, существует множество алгоритмов решения данной за-

дачи, и в работах-версиях могут использоваться способы, отличные от алгоритмов решения аналогичной задачи в работе-оригинале. К данному варианту МР можно отнести также случай, когда работа-версия и работа-оригинал написаны на различных языках программирования;

3) Резервный КВР имеет структуру, существенно отличную от структуры исходного КВР – и по числу работ, и по топологии связей; обычно имеет место при реализации принципиально различающихся версий одной и той же задачи управления (например, разработка одной и той же проблемы поручена двум конкурирующим организациям, отделам, институтам и пр.).

В работах [10, 11] было проведено исследование организации второго варианта МР при штатном выполнении КВР, т.е. в отсутствие сбоев и отказов процессоров. В данной работе впервые исследуется организация диагностирования ошибок для второго варианта МР. Данный вариант является более сложным с точки зрения организации вычислительного процесса – формирования и контроля выполнения базового (штатное выполнение) КВР и преобразованных (в условиях ошибок – сбоя или отказа процессора ВС) КВР в терминах и понятиях [2].

2. Формирование преобразованных КВР

Напомним, что основными принципами формализованного описания и организации диагностирования ошибок при выполнении КВР в параллельной ВС по [2, 12] являются стандартизация и унификация процедур формирования *преобразованных* КВР и их центрального компонента – программного *блока диагностирования* (БД): обнаружение ошибки (неисправности) процессора ВС при выполнении им любой из «тройки» контрольных работ (a_j , a_j', b_j) базового КВР – *этап 1*; определение «координаты» ошибки (номера неисправного процессора) на основе принципа асинхронной программной реализации логической функции мажорирования (голосования по большинству «два из трех») [12] – *этап 2*; определение типа неисправности процессора ВС (сбой или отказ) – *этап 3*.

На основе принципов и формальных правил по [2, 12], в [9] была разработана структура БД для первого варианта МР.

В настоящей работе авторами синтезирован новый вариант БД, модифицированный именно для рассматриваемого (второго) варианта МР при использовании подхода организации вычислительного процесса, описанного в [10].

На рис. 1 приведена функционально-логическая блок-схема контроля и диагностирования выполнения КВР в параллельной ВС, включающая предлагаемый модифицированный БД. Информационные связи между функциональными блоками изображены здесь сплошными линиями, а логические (управляющие) связи – штриховыми линиями.

Блок-схема по рис. 1 включает функциональные блоки $\tilde{A}_{j\max}^{(f)}$, $\tilde{A}_{j\min}^{(h)}$ и $\tilde{B}_j^{(d)}$, соответствующие выполнению «тройки» контрольных работ (a_j , a_j', b_j) базового КВР на некоторых процессорах $П_f$ и $П_h$, причем $f \neq h$, а $d=f$ или $d=h$ в зависимости от типа используемой диспетчеризации [10]. В данном примере для простоты примем, что $d=f$. Функциональный блок $\tilde{A}_{j\max}^{(f)}$ соответствует той из пары работ a_j , a_j', среднее время выполнения которой $M[t_j]$ больше (обозначим эту работу $a_{j\,max}$), а функциональный блок $\tilde{A}_{j\min}^{(h)}$ в свою очередь соответствует той из пары работ a_j , a_j', среднее время выполнения ко-

торой меньше (обозначим эту работу $a_{j\ min}$). Указанные блоки используются для обнаружения неисправности (этап 1 по [12]): результатами выполнения работы сравнения b_j на процессоре Π_d, т.е. выходными сигналами блока $\tilde{B}_j^{(d)}$ являются управляющие сигналы, обозначенные на рис. 5 как $Y_{bj(=)}$, $Y_{bj(\neq)}$:

$$Y_{bj(=)} = A_j^{(f)} A_j'^{(h)} B_j^{(d)}, \quad (1)$$

этот сигнал свидетельствует о штатном выполнении всех трех контрольных работ – a_j , a_j', b_j; при этом $B_j^{(d)}$=1 соответствует формированию правильного (неискаженного) результата выполнения работы сравнения b_j на процессоре Π_d ;

$$Y_{bj(\neq)} = (\overline{A}_{j\max}^{(f)} A_{j\min}^{(h)} + A_{j\max}^{(f)} \overline{A}_{j\min}^{(h)}) B_j^{(d)} + A_{j\max}^{(f)} A_{j\min}^{(h)} \overline{B}_j^{(d)}, \quad (2)$$

этот сигнал формируется либо в случае несовпадения результатов работ a_j и a_j' и правильного (неискаженного) результата выполнения работы сравнения b_j, либо в случае совпадения результатов работ a_j и a_j' и искажения результата выполнения работы сравнения b_j. Таким образом, сигнал $Y_{bj(\neq)}$=1 свидетельствует о неисправности процессора Π_f или Π_h при выполнении одной из указанных работ, и потому является сигналом инициации (запуска) блока диагностирования БД, как и в [12].

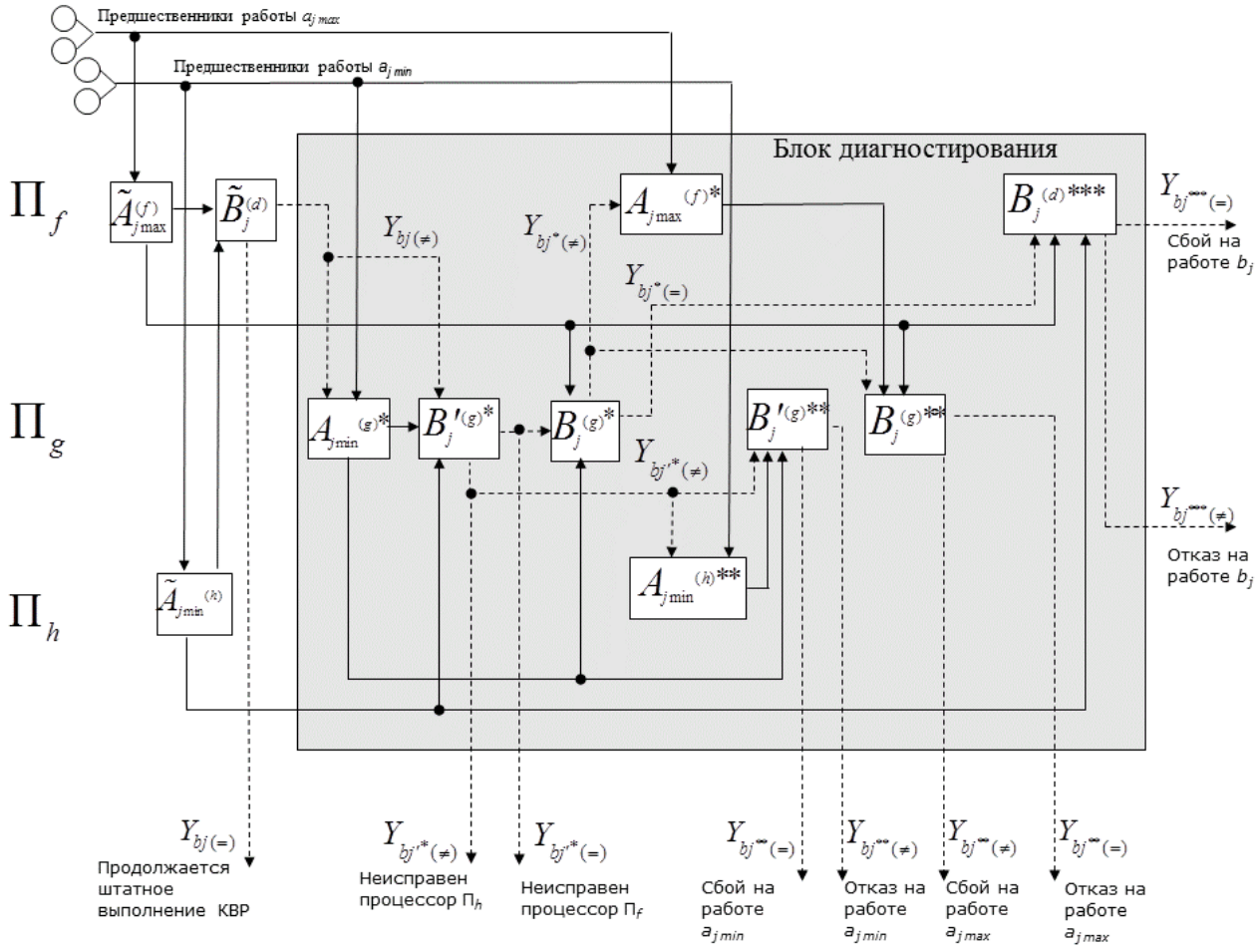

Рис. 1. Функционально-логическая блок-схема контроля и диагностирования выполнения КВР в параллельной ВС

Важная особенность нового БД, разработанного авторами именно для второй версии МР, заключается в следующем: для определения «координаты» ошибки (*этап 2* по [12]) здесь используются только «быстрые» работы $a_{j\ min}$ базового КВР и ее копия $a_{j\ min}$* – в среднем с меньшими временами реализации, чем у соответствующей им работы $a_{j\ max}$, а асинхронная программная реализация логической функции мажорирования по [12] заменена дополнительной работой сравнения b_j'* результатов двух «быстрых» работ – $a_{j\ min}$ и $a_{j\ min}$*.

Выполнению указанных работ БД – $a_{j\ min}$* и b_j'* – соответствуют функциональные блоки $A_{j\ min}^{(g)}$* и $B_j'^{(g)}$*, рис. 1, которые должны выполняться на некотором заведомо исправном процессоре $П_g$ ($g \neq f$ и $g \neq h$). Пусть обозначение функционального блока БД в квадратных скобках (например, $[A_{j\ min}^{(g)}*]$) соответствует процессу выполнения и завершения соответствующей работы $a_{j\ min}$* на процессоре $П_g$. Тогда при условии инициации БД (т.е. при $Y_{bj}(\neq)=1$) всегда имеет место $[A_{j\ min}^{(g)}*]=1$ и $[B_j'^{(g)}*]=1$.

Управляющий сигнал $Y_{bj}'^{*}_{(=)}$ (на одном из выходов блока $B_j'^{(g)}$*) формируется в случае совпадения результатов выполнения двух одинаковых копий «быстрых» работ – $a_{j\ min}$ на процессоре $П_h$ и $a_{j\ min}$* на заведомо исправном процессоре $П_g$ – при заведомо корректном выполнении дополнительной работы сравнения b_j'*. Такое совпадение свидетельствует об исправности процессора $П_h$ при выполнении работы a_{jmin} и, следовательно, об ошибке процессора $П_f$ при выполнении либо работы-оригинала $a_{j\ max}$, либо работы сравнения b_j. Формально это подтверждается следующим логическим выражением для $Y_{bj}'^{*}_{(=)}$, синтезированным по правилам [12] с учетом условия (2) включения БД:

$$Y_{bj'*(=)} = (\overline{A}_{j\max}^{(f)}\ B_j^{(d)} + A_{j\max}^{(f)}\ \overline{B}_j^{(d)})(A_{j\ min}^{(h)}\ [A_{j\ min}^{(g)*}]). \tag{3}$$

Аналогично, НЕсовпадение результатов указанных работ – $a_{j\ min}$ и $a_{j\ min}$* – свидетельствует об ошибке процессора $П_h$, на котором выполнялась работа-версия $a_{j\min}$ базового КВР, и, наоборот, об исправности процессора $П_f$, что подтверждается синтезированным (по правилам [12]) логическим выражением для управляющего сигнала $Y_{bj}'^{*}_{(\neq)}$ (на другом выходе блока $B_j'^{(g)}$*):

$$Y_{bj'*(\neq)} = (A_{j\max}^{(f)}\ B_j^{(d)})(\overline{A}_{j\min}^{(h)}\ [A_{j\ min}^{(g)*}]). \tag{4}$$

Таким образом, с помощью двух функциональных блоков БД – $A_{j\ min}^{(g)}$* и $B_j'^{(g)}$*– однозначно определяется «координата» ошибки (*этап 2* по [12]) – номер неисправного процессора ($П_f$ либо $П_h$), на котором обнаружена неисправность (сбой или отказ процессора) при выполнении какой-либо из контрольных работ (a_j , a_j', b_j) базового КВР.

Напомним по [12], что присутствие в выходных функциях БД хотя бы одного логического компонента с логическим отрицанием (например, в выражении (3) – компонента $\overline{A}_{j\max}^{(f)}$ для выполнения работы a_j на процессоре $П_f$) соответствует неисправности процессора ($П_f$), указанного в компоненте.

Для более точного определения «координаты» ошибки – идентификации работы в последовательности $a_{j\ max}$, b_j в случае неисправности выполнявшего их процессора $П_f$, – в БД включен функциональный модуль $B_j^{(g)}$* (рис. 5), соответствующий выполнению дополнительной работы сравнения b_j* – копии работы сравнения b_j базового КВР – на заведомо исправном процессоре $П_g$, и потому выполнение работы b_j* заведомо полага-

ется корректным. Эта работа предназначена для сопоставления результатов выполнения работы $a_{j\ max}$ базового КВР и копии «быстрой» работы $a_{j\ min}$*. (Напомним, что результаты последней заведомо полагаются правильными, т.е. неискаженными). При совпадении результатов работ $a_{j\ max}$ и $a_{j\ min}$* на одном из выходов функционального блока $B_j^{(g)}$* формируется управляющий сигнал $Y_{bj*(=)}=Y_{bj'*(=)}(A_j^{(f)}[A_j'^{(g)}*])$, а с учетом выражения (3) для сигнала $Y_{bj'*(=)}$, «указывающего» на неисправность процессора Π_f имеем:

$$Y_{bj*(=)} = A_{j\max}^{(f)}\ A_{j\max}^{(f)}\ A_{j\ min}{}^{(h)}\ [A_{j\ min}{}^{(g)*}], \tag{5}$$

что свидетельствует об ошибке процессора Пf при выполнении именно работы сравнения b_j базового КВР.

При НЕсовпадении результатов работ $a_{j\ max}$ и $a_{j\ min}$*, на другом выходе функционального блока $B_j^{(g)}$* формируется (с учетом того же выражения (3) для $Y_{bj'*(=)}$) управляющий сигнал:

$$Y_{bj*(\neq)} = \overline{A}_{j\max}^{(f)}\ A_{j\ min}{}^{(h)}\ B_j^{(d)}\ [A_{j\ min}{}^{(g)*}], \tag{6}$$

этот сигнал «указывает» на ошибку процессора Пf при выполнении именно работы $a_{j\ max}$ базового КВР.

Для реализации *этапа 3 (*определение типа неисправности – *сбой* либо *отказ* – процессора, на котором зафиксировано искажение результатов вычислений одной из контрольных работ базового КВР) используется алгоритмический прием, известный еще для обнаружения сбоев процессоров даже в последовательных, однопроцессорных ВС: один и тот же программный модуль (или фрагмент программы между двумя контрольными точками) выполняется не менее чем два раза на одном и том же (в последовательных ВС – на единственном) процессоре.

Так, для определения типа неисправности процессора Π_h, зафиксированной при выполнении «быстрой» работы $a_{j\ min}$ базового КВР, на тот же процессор Π_h назначается дополнительная копия $a_{j\ min}$** работы $a_{j\ min}$ базового КВР (выполнению работы $a_{j\ min}$ ** соответствует функциональный блок $A_{j\ min}{}^{(h)}$** БД, рис. 1); с помощью дополнительной работы сравнения b_j'**, реализуемой на заведомо исправном процессоре Π_g (функциональный блок $B_j'^{(g)}$**, рис. 1), результаты выполнения работы $a_{j\ min}$** сопоставляются с заведомо правильными результатами работы $a_{j\ min}$* (функциональный блок $A_{j\ min}{}^{(g)}$* на рис. 1). В случае совпадения результатов этих работ, что соответствует сбою процессора Π_h при выполнении работы $a_{j\ min}$ базового КВР на этом процессоре, на одном из выходов функционального блока $B_j'^{(g)}$** формируется (с учетом выражения (6)) сигнал $Y_{bj'**(=)}$:

$$Y_{bj'**(=)} = (A_{j\max}^{(f)}\ \overline{A}_{j\min}{}^{(h)}\ B_j^{(d)})([A_{j\ min}{}^{(g)*}]\ [A_{j\ min}{}^{(h)**}]), \tag{7}$$

а в случае НЕсовпадения этих результатов, на другом выходе $B_j'^{(g)**}$ формируется сигнал:

$$Y_{bj'**(\neq)} = (A_{j\max}^{(f)}\ \overline{A}_{j\min}{}^{(h)}\ B_j^{(d)})([A_{j\ min}{}^{(g)*}]\ [\overline{A}_{j\min}{}^{(h)**}]), \tag{8}$$

свидетельствующий об отказе процессора Π_h при выполнении работы $a_{j\ min}$.

Рассмотрим процедуру определения типа неисправности процессора Π_f, зафиксированной при выполнении именно «долгой» работы $a_{j\,max}$ базового КВР, – на такой случай «указывает» управляющий сигнал $Y_{bj}*_{(\neq)}=1$, выражение (6), являющийся одним из результатов выполнения функционального блока $B_j^{(g)}*$ (рис. 1). По этому сигналу на тот же процессор Π_f немедленно назначается копия $a_{j\,max}*$ работы $a_{j\,max}$ базового КВР (выполнению работы $a_{jmax}*$ соответствует функциональный блок $A_{j\,max}^{(f)}*$, рис. 1).

С помощью дополнительной работы сравнения b_j**, реализуемой на заведомо исправном процессоре Π_g (функциональный блок $B_j^{(g)}**$, рис. 1), результаты работы $a_{j\,max}*$ сравниваются с результатами работы-оригинала $a_{j\,max}$ (функциональный блок $A_{j\,max}^{(f)}$ на рис. 1). В случае совпадения результатов этих работ (что соответствует *отказу* процессора Π_f при выполнении работы $a_{j\,max}$ базового КВР на этом процессоре), на одном из выходов функционального блока $B_j^{(g)}**$ формируется (с учетом (6)) сигнал $Y_{bj}**_{(=)}$:

$$Y_{bj**(=)} = (\overline{A}_{j\max}^{(f)}\ A_{j\min}^{(h)}\ B_j^{(d)})\ ([A_{j\min}^{(g)*}][\overline{A}_{j\max}^{(f)*}]), \tag{9}$$

а в случае НЕсовпадения указанных результатов, на другом выходе блока $B_j^{(g)}**$ формируется управляющий сигнал:

$$Y_{bj**(\neq)} = (\overline{A}_{j\max}^{(f)}\ A_{j\min}^{(h)}\ B_j^{(d)})\ ([A_{j\min}^{(g)*}][A_{j\max}^{(f)*}]), \tag{10}$$

свидетельствующий о *сбое* процессора Π_f при выполнении работы $a_{j\,max}$.

Рассмотрим событие ошибки процессора Π_f при выполнении работы сравнения b_j базового КВР, о котором свидетельствует сигнал $Y_{bj*(=)}$ с выхода блока $B_j^{(g)}*$ (рис. 1). В этом случае на процессор Π_f назначается дополнительная работа сравнения b_j*** – копия работы b_j, заново сверяющая результаты работ $a_{j\,max}$ и $a_{j\,min}$. В случае сбоя процессора Π_f при выполнении работы b_j, на одном из выходов соответствующего функционального блока $B_j^{(d)}***$ формируется управляющий сигнал:

$$Y_{bj***(=)} = \overline{B}_j^{(d)}(A_{j\max}^{(f)}\ A_{j\min}^{(h)}[A_{j\min}^{(g)*}][B_j^{(d)***}]), \tag{11}$$

а в случае отказа процессора Π_f при выполнении работы b_j, на другом выходе того же блока формируется сигнал:

$$Y_{bj***(\neq)} = \overline{B}_j^{(d)}(A_{j\max}^{(f)}\ A_{j\min}^{(h)}[A_{j\min}^{(g)*}][\overline{B}_j^{(d)***}]). \tag{12}$$

Альтернативные управляющие сигналы (5) – (12) используются для определения подмножеств $\{a_l\}$ *работ отката* [12], которые необходимо перезапускать для восстановления вычислительного процесса.

При обнаружении ошибки (искажения данных) во время выполнения работ базового и преобразованного КВР на одном из процессоров (Π_f или Π_h), т.е. *отказа*, данный процессор должен быть исключен из дальнейших вычислений.

Авторы считают, что в данной ситуации целесообразным является изменение способа организации вычислительного процесса: по завершении выполнения множества

работ отката $\{a_l\}$ все работы сравнения b_j назначаются на выполнение на специально выделенный процессор.

При использовании парных ресурсов (или четного количества вычислителей ВС) в качестве выделенного может быть выбран процессор, находящийся в паре с отказавшим (и исключенным из вычислительного процесса).

Т.е. фактически в *динамике* вычислений происходит переход к подходу организации вычислительного процесса, который базируется на структурной избыточности вычислителей ВС. Данный поход был рассмотрен в [11].

3. Заключение

Для рассматриваемого варианта МР впервые формально обоснована, синтезирована и логически описана новая структура программного блока диагностирования (БД) для определения «координаты» одиночной неисправности (номера неисправного процессора ВС), ее типа (сбой или отказ) и идентификации работы КВР (программного модуля) с искаженными результатами ее выполнения.

Также рассмотрен один из способов организации сочетания и алгоритм взаимодействия двух подходов [10, 11] к реализации рассматриваемого в данной работе варианта МР, как перспективных компонентов динамического адаптивного резервирования взаимосвязанных программных модулей [13].

СПИСОК ЛИТЕРАТУРЫ

1. *Елисеев В.В., Игнатущенко В.В.* О проблеме надежного выполнения сложных наборов задач в управляющих параллельных вычислительных системах // Проблемы управления. 2006. № 6. С. 6–18.
2. *Игнатущенко В.В., Исаева Н.А.* Резервирование взаимосвязанных программных модулей для управляющих параллельных вычислительных систем: организация, оценка отказоустойчивости, формализованное описание // Автоматика и телемеханика. 2008. № 10. С. 142 –161.
3. *Игнатущенко В.В., Исаева Н.А.* Интеллектуальное динамическое управление параллельными резервированными взаимосвязанными задачами со случайными временами их выполнения в управляющих параллельных вычислительных системах // Труды пятой международной конференции «Параллельные вычисления и задачи управления» (PACO’2010). М.: Учреждение Российской академии наук Институт проблем управления им. В.А. Трапезникова РАН. 2010. С. 643 –651.
4. *Авиженис А.* Отказоустойчивость – свойство, обеспечивающее постоянную работоспособность цифровых систем // ТИИЭР. Т.66. 1978. № 10. С. 5–25.
5. *Головкин Б.А.* Параллельные вычислительные системы. М.: Наука.1980. 519с.
6. *Laura Pullum.* Software fault tolerance techniques and implementation // Artech House. 2001. 343с.
7. *Игнатущенко В.В., Исаева Н.А., Подшивалова И.Ю.* Проблема адаптивного резервирования вычислительных процессов для их надежного выполнения в управляющих параллельных вычислительных системах // Третья международная конференция по проблемам управления: Пленарные доклады и избранные труды. М.: Институт проблем управления РАН. 2006. С. 775 – 781.

8. *Исаева Н.А.* Унифицированная математическая модель системы массового обслуживания для прогнозирования надежного выполнения взаимосвязанных программных модулей (задач) при различных методах и сочетаниях их резервирования в управляющих параллельных вычислительных системах // Открытое образование. 2011. № 2 (85). Ч. 2. С. 55–59.
9. *Игнатущенко В.В., Милков М.Л., Сидоров А.В.* Многоверсионное резервирование взаимозависимых параллельных задач для управляющих параллельных вычислительных систем: формализованное описание, оценка отказоустойчивости // Надежность. 2009. № 4. С. 44–61.
10. *Милков М.Л.* Исследование способов диспетчеризации работ для одного из вариантов многоверсионного резервирования взаимосвязанных программных модулей в управляющих параллельных вычислительных системах // Открытое образование. 2011. № 2 (85). Ч. 2. С. 48–52.
11. *Исаева Н.А., Милков М.Л.* Организация многоверсионного резервирования взаимосвязанных задач со случайными временами их выполнения в управляющих параллельных вычислительных системах реального времени: новый подход // Труды пятой международной конференции "Управление развитием крупномасштабных систем (MLSD'2011)" (3-5 октября 2011 г., Москва, Россия). Том II. М.: Учреждение Российской академии наук Институт проблем управления им. В.А. Трапезникова РАН. 2011. С. 152–157.
12. *Исаева Н.А.* Логический синтез процедур резервирования взаимосвязанных программных модулей в параллельных вычислительных системах // Труды XXXV Междунар. Конф. «Информационные технологии в науке, социологии, экономике и бизнесе» (IT+SE’08). Май 2008. Украина. Ялта-Гурзуф. С. 64–68.
13. *Исаева Н.А.* Организация динамического адаптивного резервирования взаимосвязанных программных модулей в управляющих параллельных вычислительных системах реального времени // Труды пятой международной конференции "Управление развитием крупномасштабных систем (MLSD'2011)" (3-5 октября 2011 г., Москва, Россия). Том II. М.: Учреждение Российской академии наук Институт проблем управления им. В.А. Трапезникова РАН. 2011. С. 230–238.