УДК 004.05

А.М. РОМАНКЕВИЧ, МОРАВЕДЖ СЕЙЕД МИЛАД, В.А. РОМАНКЕВИЧ, К.В. МОРОЗОВ

Национальный технический университет Украины «КПИ», Украина

ОБ ОДНОЙ ЗАДАЧЕ РЕКОНФИГУРИРОВАНИЯ В МНОГОПРОЦЕССОРНЫХ СИСТЕМАХ

Рассматриваются вопросы организации реконфигурирования в многопроцессорных системах управления на разных стадиях их проектирования и функционирования. Предлагается алгоритм уменьшения перебора при определении размещения функций управления по процессорам системы, основой которого является таблица, которая трансформируется по мере появления отказов процессоров. В основу алгоритма положена таблица возможных размещений функций по процессорам, которая последовательно трансформируется по мере появления отказов. Приводится алгоритм расчета числа реальных размещений.

Ключевые слова: *реконфигурирование, многопроцессорные системы, системы управления.*

Введение

Многопроцессорные системы (МС) [1] имеют структурную и временную избыточность, способны сами себя тестировать, обнаруживать и исключать из функционирования вышедшие из строя процессоры, реконфигурироваться и продолжать работу.

Любая система управления каким-то объектом выполняет некоторое множество $F=\{f_1,f_2,\dots f_N\}$ функций управления. Определенное подмножество $F_{оп} \subset F$ этих функций, невыполнение хотя бы одной из которых вызывает появление опасного состояния объекта управления, называют опасным, а функции из $F_{оп}$ - опасными. Для уточнения понятий, которыми мы оперируем, скажем следующее. Следуя основным канонам теории гарантоспособности (см., например, [2]), можно отметить, что система может находиться в одном из нескольких состояний: работоспособное (выполняются все функции из F, в том числе, когда МС является отказоустойчивой [3]), частично работоспособное (выполняются функции из некоторого подмножества функций $F^* \subset F$, но $F_{оп} \subset F^*$), полностью неработоспособное (может быть опасным и безопасным).

Среди особенностей МС большое значение имеет возможность реконфигурирования: после выполнения процедуры взаимного тестирования процессоров и установления процессора, вышедшего из строя, оставшиеся работоспособными берут на себя функции, которые он выполнял до отказа.

1. Ограничения

Следует отметить, что в многопроцессорных системах управления иногда имеются различные ограничения на распределение функций по процессорам. Понятно, что каждая функция для своей реализации требует какого-то количества операций в единицу времени, то есть какой-то части производительности процессора, что можно обозначить как S_{fi}. Если S_p – производительность процессора, то первое, вполне очевидное ограничение для каждого процессора:

$$S_p \geq \sum S_{fi}$$

для множества функций, которые погружаются в данный процессор.

Второе ограничение: каждая функция может быть выполнена в одном из процессоров некоторого подмножества процессоров системы. Это ограничение может быть вызвано различными причинами: особенности и возможности процессора и его программного обеспечения, наличие или отсутствие каналов связи с источниками информации извне и др.

2. Алгоритм

Возьмем за основу следующий критерий выбора множества функций, которые остаются для исполнения: максимально возможное количество функций при максимально возможной суммарной производительности, которую они требуют для своего выполнения (максимально приближенной к суммарной производительности оставшихся работоспособными процессоров).

Задача, рассматриваемая в настоящей работе, собственно, и состоит в оптимизации выбора распределения функций по процессорам при наличии указанных ограничений.

Задача, скорее всего, не имеет прямого алгоритмического решения, то есть какой-то перебор должен иметь место. Пока число возможных реальных отказов менее степени отказоустойчивости сис-

темы существует достаточно много решений, так как имеется определенная избыточность. Затем, по мере появления отказов приходится решать задачу оптимизации процесса реконфигурирования согласно принятому критерию при недопущении (пока это возможно) попадания системы в опасное состояние.

Пусть имеется многопроцессорная система, содержащая n процессоров, причём система должна выполнять N функций управления, и имеются указанные ограничения. В примере, рассмотренном ниже, n=6 и N=10, причем функция f_1 может быть выполнена либо процессором P_{r1}, либо процессором P_{r2}, функция f_2 –одним из процессоров P_{r1}, P_{r2}, P_{r5} или P_{r6} и т.д., в соответствии с табл. 1. Каждый процессор имеет свою производительность S_{pi}, каждая функция требует для своего выполнения - S_{fi}.

Алгоритм начинается с построения таблицы, подобной табл. 1. В ней строки обозначены N_i – подмножества функций, которые способен выполнять i-й процессор, столбцы указывают подмножество процессоров, куда функция f_i может быть помещена. Для простоты полагаем, что все процессоры имеют одинаковую производительность (хотя это не обязательно), равную 6 условных единиц. Нижняя строка таблицы – производительность (в тех же условных единицах), которая требуется для выполнения соответствующей функции.

Далее определяется $S^*= \sum S_{fi}$ для функций, которые подлежат исполнению, а также $S = \sum S_{pi}$ для процессоров, которые являются работоспособными. Для отказоустойчивых МС до появления отказов распределение функций не составляет труда, поскольку имеется избыточность, и $S > S^*$.

Таблица 1

Исходная таблица
распределения функций управления

	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}
N_1	*	*		*	*		*			
N_2		*			*	*			*	*
N_3	*					*		*	*	
N_4			*	*			*			*
N_5		*	*			*	*			
N_6		*			*		*	*		*
	3	2	3	4	2	3	3	3	3	4

Далее, после появления очередного отказа строится новая таблица (без соответствующей строки), и по ней определяется новое распределение функций. Некоторое облегчение в выборе появляется, если в столбце имеется единственная отметка.

Если распределение оказывается невозможным, например, когда $S^*>S$, то приходится исключать из множества F некоторое подмножество σ функций согласно выбранному критерию. При этом $\sigma \cap F_{оп} = \varnothing$. Строится новая таблица путем вычеркивания столбцов, соответствующих функциям из σ, для которой вновь подсчитываются S и S*, и производится поиск распределения, то есть выбираются такие комбинации по строкам, чтобы:

- каждая функция участвовала лишь один раз;
- сумма весов функций, оставшихся в каждой строке, не превышала производительности соответствующего процессора.

Если поиск не дает результата, приходится выбирать другое подмножество σ, строить другую таблицу, и процесс повторяется. То же самое происходит при обнаружении каждого последующего отказа. По мере появления отказов процессоров таблица уменьшается, и перебор вариантов выбора σ и распределения функций сокращается. Конец наступает тогда, когда условие $\sigma \cap F_{оп} = \varnothing$ становится невыполнимым.

Для нашего примера исходным может быть, например, следующее распределение:

$$N_1=\{f_4,f_5\};\ N_2=\{f_{10},f_2\};\ N_3=\{f_1,f_9\};$$
$$N_4=\{f_3\};\ N_5=\{f_6\};\ N_6=\{f_7,f_8\}.$$

Пусть процессор P_{r4} выходит из строя. Если вычеркнуть строку N_4 из таблицы, то легко увидеть, что f_3 может быть передана в N_5, и при этом ограничения соблюдаются. Пусть далее выходит из строя процессор P_{r6}. Поскольку величина S уменьшается до 24, то S* также не должна превышать этой величины. Если $F_{оп}=\{f_7,f_2\}$, то выбирать $\sigma=\{f_7,f_8\}$ нельзя. Выбираем $\sigma=\{f_6,f_8\}$, и распределение остается прежним, если в N_5 поместить f_7 вместо f_6. Остается 4 процессора и, если далее выходит из строя P_{r2}, то трансформированная таблица приобретает следующий вид (табл. 2; промежуточные таблицы опущены).

Таблица 2

Трансформированная таблица
распределения функций управления

	f_1	f_2	f_3	f_4	f_5	f_7	f_9	f_{10}
N_1	*	*		*	*	*		
N_3	*						*	
N_5		*	*			*		

Ясно, что новое множество σ должно включать f_{10} и, чтобы S* максимально было приближено к S=18, добавляем туда f_5 , то есть $\sigma=\{f_5,f_{10}\}$. Распределение оставшихся функций очевидно:

$$N_1=\{f_4,f_2\};\ N_3=\{f_1,f_9\};\ N_5=\{f_3,f_7\}.$$

Понятно также и распределение функций при появлении последующего отказа: если это P_{r1} или P_{r5}, то функции f_7 и f_2 помещаются в оставшийся из них работоспособным (так как обе входят в $F_{оп}$), и система может оставаться в состоянии частичной работоспособности до тех пор, пока он не выйдет из строя.

Можно отметить, что исходная таблица иногда может помочь разработчику в плане уточнения возможной степени отказоустойчивости системы. В примере система не может быть даже 1-отказоустойчивой, в частности, потому, что отказ P_{r6} исключает возможность выполнения всего множества функций, поскольку не существует нужного распределения из-за ограничения 2.

3. Оценка числа возможных вариантов распределения

Пример 1 показывает, что распределение функций по процессорам может быть далеко не единственным. Количество возможных вариантов распределения может служить хорошим критерием при выборе множества σ. Возникает задача предварительного определения числа возможных распределений q без перебора всех возможных комбинаций. Учитывая сложность ее решения, особенно при больших значениях n и N, рассмотрим алгоритм определения величины q для более простого случая, когда значения S_{pi} (а также S_{fi}) одинаковы для всех i. Идея алгоритма заключается в подсчете числа всех возможных комбинаций распределения с учетом ограничения 2, из которого затем вычитается число всех комбинаций, запрещенных в силу ограничения 1.

Выполнение алгоритма будем иллюстрировать следующим примером.

Пример 2. Пусть n=4 и N=8 и имеет место возможное распределение (ограничение 2):

$$N_1=\{f_1,f_2,f_3\};\ N_2=\{f_2,f_5,f_6,f_7\};$$
$$N_3=\{f_3,f_4,f_6,f_8\};\ N_4=\{f_5,f_7,f_8\}.$$

Ограничение 1: каждый процессор может выполнять 2 функции.

1. Вначале определяем общее количество Q теоретически возможных комбинаций распределения при отсутствии ограничений 1 и 2. Если обозначить через a_i число возможных вариантов помещения функций f_i, то легко увидеть:

Q=Πa_i, для нашего примера Q=64.

2. Подсчитываем число возможных комбинаций распределения функций по всем процессорам с учетом лишь ограничения 2 для всех теоретически возможных значений величины S_p, начиная с максимального и заканчивая на единицу большим допустимого. Фактически эти комбинации являются запрещёнными, поскольку не соответствуют ограничению 1.

2.1 Для нашего примера это выглядит следующим образом.

Для S = 4 (максимальное число функций в одном процессоре)

Pr_2 $f_2f_5f_6f_7 \Rightarrow f_1(1)f_3(2)f_4(1)f_8(2) \Rightarrow 4$.

Здесь слева перечислены функции, помещенные в Pr_2, справа – оставшиеся функции, которые могут быть помещены в другие процессоры произвольно. Число в скобках – количество таких возможных размещений каждой функции. Цифра 4 справа – число комбинаций размещения функций, если в процессоре Pr_2 находятся функции f_2, f_5, f_6 и f_7 : 1×2×1×2 = 4.

Pr_3 $f_3f_4f_6f_8 \Rightarrow f_1(1)\ f_2(2)\ f_5(2)\ f_7(2) \Rightarrow 8$.

Общее число запрещенных комбинаций (для S = 4): $R_4 = 12$.

Далее определяем число запрещенных комбинаций для S = 3.

Pr_1 $f_1f_2f_3 \Rightarrow f_4(1)f_5(2)f_6(2)f_7(2)f_8(2) \Rightarrow 16$.

Pr_2 $f_2f_5f_6 \Rightarrow f_1(1)f_3(2)f_4(1)f_7(1)f_8(2) \Rightarrow 4$.
$f_2f_5f_7 \Rightarrow f_1(1)f_3(2)f_4(1)f_6(1)f_8(2) \Rightarrow 4$.
$f_2f_6f_7 \Rightarrow f_1(1)f_3(2)f_4(1)f_5(1)f_8(2) \Rightarrow 4$.
$f_5f_6f_7 \Rightarrow f_1(1)f_2(2)f_3(2)f_4(1)f_8(2) \Rightarrow 4$.

Pr_3 $f_3f_4f_6 \Rightarrow f_1(1)\ f_2(2)\ f_5(2)\ f_7(2)\ f_8(1) \Rightarrow 8$.
$f_3f_4f_8 \Rightarrow f_1(1)\ f_2(2)\ f_5(2)\ f_6(1)\ f_7(2) \Rightarrow 8$.
$f_3f_6f_8 \Rightarrow f_1(1)\ f_2(2)\ f_4(1)\ f_5(2)\ f_7(2) \Rightarrow 8$.
$f_4f_6f_8 \Rightarrow f_1(1)\ f_2(2)\ f_3(1)\ f_5(2)\ f_7(2) \Rightarrow 8$.

Pr_4 $f_5\ f_7\ f_8 \Rightarrow f_1(1)\ f_2(2)\ f_3(2)\ f_4(1)\ f_6(2) \Rightarrow 8$.

Всего 72 комбинации для S=3, а общее число запрещенных комбинаций R= 84.

2.2 Поскольку очевидно, что имеют место повторы учёта одних и тех же комбинаций, подсчитываем их. В нашем примере вначале определяем число повторов, связанных с присутствием в какомлибо процессоре 4-х функций $P_{S=4}$. Их всего 2: $f_2\ f_5\ f_6\ f_7$ в Pr_2 и $f_3f_4f_8$ в Pr_3, а также $f_3f_4f_6f_8$ в Pr_3 и $f_2f_5f_7$ в Pr_2.

Анализируя комбинации из 3-х функций в одном процессоре, отмечаем 8 повторов, связанных с $f_3f_6f_8$ в Pr_3, так как функция f_4 (одна из недостающих) может быть помещена только в этом процессоре, то есть любой вариант распределения всех 8-ми функций, который включает в себя размещение f_3,f_6,f_8 в Pr_3, содержит там же и f_4, а это уже учтено при анализе S=4.

Далее, имея все возможные комбинации из 3-х функций, которые могут быть помещены в тот или иной процессор, рассмотрим возможные комбинации, которые включают по 3 функции в каких-либо 2-х процессорах одновременно, но исключая те, которые так или иначе уже были учтены ранее при анализе случая S=4.

Число повторов также легко определяется по возможным комбинациям недостающих функций, например,

$$f_1f_2f_3 - f_5f_6f_7 \Rightarrow f_4(1)\ f_8(2) \Rightarrow 2.$$

Продолжая рассуждения аналогично приведенным, получаем числа других повторов разных комбинаций. Их следует перебрать полностью.

Выполнив эту процедуру, находим количество повторов, связанных с S=3: $P_{S=3}$=8+15=23. Общее количество повторов $P = P_{S=3} + P_{S=4}$= 25.

2.3 Следовательно, число запрещенных неповторяющихся комбинаций

$$Q^* = R - P = 59.$$

3. Определяем число разрешенных комбинаций: $q = Q - Q^*$.

Для нашего примера это, действительно, всего 5 комбинаций (64 – 59), и одна из них, например, следующая: в Pr_1 - (f_1f_2), в Pr_2 - (f_5f_6), в Pr_3 - (f_3f_4), в Pr_4 - (f_7f_8).

Выводы

В работе предложен алгоритм сокращения перебора при выборе функций при их распределении по процессорам в процессе реконфигурации многопроцессорной системы управления с учетом определенных ограничений. Оптимизация этого выбора может быть связана с достаточно большим перебором, и его сокращение представляет определенный интерес. В основу алгоритма положена таблица возможных размещений функций по процессорам, которая последовательно трансформируется по мере появления отказов. Приводится алгоритм расчета числа реальных размещений.

Литература

1. Реконфигурируемые мультиконвейерные вычислительные структуры [Текст] / И.А. Каляев, И.И. Левин, Е.А. Семерников, В.И. Шмойлов; под общ. ред. И.А.Каляева. – Ростов / Д: ЮНЦ РАН. – 2008. – 320 с.

2. Харченко, В.С. Гарантоспособность и гарантоспособные системы: элементы методологии [Текст] / В.С. Харченко // Радиоэлектронные и компьютерные системы. – 2006. – № 5 (17). – С. 7 – 19.

3. Романкевич, А.М. О повышении надёжности реконфигурируемых отказоустойчивых систем управления сложными объектами [Текст] / А.М. Романкевич, В.А. Романкевич, С.М. Мораведж // Электронное моделирование. – 2010. – Т.32, № 4. – С. 85 – 92.

Поступила в редакцию 12.02.2009

Рецензент: д-р техн. наук, проф. И.А. Фурман, Харьковский национальный технический университет сельского хозяйства им. Петра Василенко, Харьков, Украина.

ПРО ОДНУ ЗАДАЧУ РЕКОНФІГУРУВАННЯ У БАГАТОПРОЦЕСОРНИХ СИСТЕМАХ

О.М. Романкевич, Мораведж Сейед Мілад, В.О. Романкевич, К.В. Морозов

Розглядаються питання організації реконфігурування в багатопроцесорних системах керування на різних стадіях їх проектування і функціонування. Пропонується алгоритм зменшення перебору при визначенні розміщення функцій керування по процесорах системи, основою якого є таблиця,яка трансформується в міру появи відмов процесорів. У основу алгоритму покладена таблиця можливих розміщень функцій по процесорах, яка послідовно трансформується у міру появи відмов. Приводиться алгоритм розрахунку числа реальних розміщень.

Ключові слова: реконфігурування, багатопроцесорні системи, системи керування.

ABOUT ONE PROBLEM OF RECONFIGURATION IN MULTIPROCESSOR SYSTEMS

A.M. Romankevich, Moravej Seyed Milad, V.A. Romankevich, K.V. Morozov

Some questions of organization of multiprocessor control systems' reconfiguration at its various degign and functioning stages are reviewed. An algorithm for decreasing of recombination under finding of control functions' distribution on the system's processors based on the table which being transformed in accordance with appearance of the processors' fault is proposed. In basis of algorithm the table of the possible placing of functions is fixed on processors, which is consistently transformed as far as appearance of refuses. An algorithm over of calculation of number of the real placing is brought.

Key words: reconfiguration, multiprocessor systems, control systems.

Романкевич Алексей Михайлович – д-р техн. наук, проф., проф. кафедры специализированных компьютерных систем Национального технического университета Украины «КПИ», Киев, Украина, e-mail: romankev@scs.ntu-kpi.kiev.ua.

Мораведж Сейед Милад – аспирант кафедры специализированных компьютерных систем Национального технического университета Украины «КПИ», Киев, Украина, e-mail: romankev@scs.ntu-kpi.kiev.ua.

Романкевич Виталий Алексеевич – канд. техн. наук, доцент, доцент кафедры Специализированных компьютерных систем Национального технического университета Украины «Киевский политехнический институт», Киев, Украина, e-mail: romankev@scs.ntu-kpi.kiev.ua.

Морозов Константин Вячеславович – магистрант кафедры Специализированных компьютерных систем Национального технического университета Украины «КПИ», Киев, Украина, e-mail: romankev@scs.ntu-kpi.kiev.ua.