2.5. ТЕХНИКА ЗА ВГРАЖДАНЕ НА НЕДЕТЕРМИНИЗЪМ В ОПЕРАТОРА *SELECT*

Недетерминизмът е фундаментално понятие, срещано в теорията на автоматите, алгоритмите и паралелната обработка [17]. Той е вътрешно присъщ за паралелните системи и в тази област се разглежда за пръв път от Dijkstra. Въвеждането на защитените команди от Dijkstra позволява този недетерминизъм не само да се овладее, но и да се използва за намирането на ефективни паралелни решения [25]. Защитените команди на Dijkstra са развити от Hoare в теорията на взаимодействащите последователни процеси *CSP* [16, 19, 31].

Смяната на изчислителния модел от последователен към паралелен не се проявява единствено с навлизането на структурния паралелизъм (под формата на многоядрени и многопроцесорни машини). Постепенно се налага архитектурната поддръжка на паралелизма – паралелното изпълнение на множество процеси и взаимодействието им се осигурява не от операционната система, а на машинно ниво.

В теорията на автоматите се разграничават детерминирани и недетерминирани автомати. С всеки детерминиран автомат е свързана функцията на преходите *F*: (*A x S*) → *S*, където *A* е азбуката, а *S* - множеството състояния на автомата. Вход за функцията на преходите се явява даден елемент от азбуката *A* и конкретно състояние от множеството *S*. Изход се явява състояние от *S*. При детерминираните автомати, за всеки вход има точно определен изход. Ако *F* не е функция, а непълно множество правила, автоматът се определя като недетерминиран [2, 17].

Паралелната обработка предполага друга интерпретация на недетерминизма – той се проявява в неопределеността, в случайния характер на избора на прехода, а не в отсъствието на предварително дефинирани преходи.

Нека е дадена следната защитена команда[1] с две алтернативи

$$\{G_1 \rightarrow P_1 \square G_2 \rightarrow P_2\}, \tag{2.2}$$

1 *Защитената команда* (guarded command) е известна още като алтернативна команда или команда за недетерминиран избор.

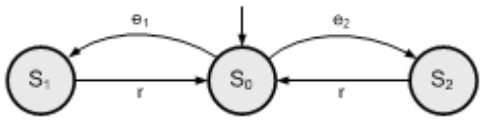
където G_1 и G_2 са съответно защитите на процесите P_1 и P_2. Ако се предположи, че защитите не сработват едновременно, т.е. събитията e_1 и e_2 настъпват поотделно, се получава диаграмата от фиг. 2.12.

Ако обаче събитията e_1 и e_2 настъпят едновременно, изборът на алтернатива е случаен – ще се избере или процеса P_1 и ще се премине в S_1, или ще се избере процеса P_2 и ще се премине в S_2. След връщането в изходното състояние ще се отработи и другото събитие, тъй като неговата защита все още ще бъде в истина.

Подобна на ситуацията от фиг. 2.13 се получава, ако защитите са команди за взаимодействие, например

$$\{P_1?x \rightarrow SKIP \square P_2?x \rightarrow SKIP\}. \tag{2.3}$$

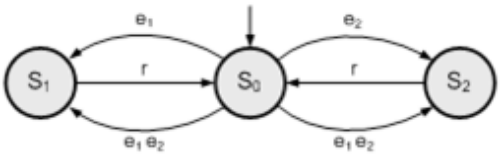
В този случай, събитието e_1 отговаря на получаването на съобщение от процеса P_1, а събитието e_2 - на получаването на



Фиг. 2.12. Диаграма на състоянията при детерминиран избор

съобщение от процеса P_2. Вследствие на паралелизма на ниво данни (DLP, [42]), двете събития могат да възникнат едновременно.

Командата на *CSP* за алтернативен избор има няколко известни реализации – операторите *ALT* в езика *OCCAM*, *select* в езика *Ada*



Фиг. 2.13. Диаграма на състоянията при недетерминиран избор

[22] и в езика *XC* [49, 53]. Всяка една реализация може да се провери за поддръжка на съдържащия се по дефиниция в алтернативната команда на *CSP* недетерминизъм.

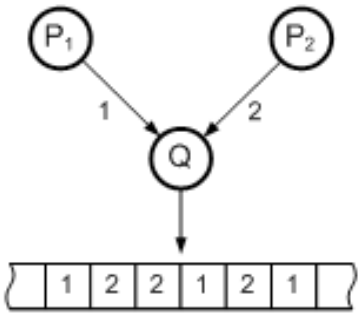
Опитната постановка представлява паралелна система от три процеса – два източника P_1 и P_2 и един консуматор Q на съобщенията (фиг. 2.14). Описва се от *CSP* уравненията

$$\{P_1||P_2||Q\},$$
$$P_1 = P_2 = P =* \{Q!msg \rightarrow \Delta \rightarrow SKIP\}_L, \tag{2.4}$$
$$Q =* \{P_1?x \rightarrow write(x) \rightarrow SKIP \square P_2?x \rightarrow write(x) \rightarrow SKIP\}_{2\times L}.$$

Източниците P_1 и P_2 изпълняват по L на брой цикъла. В рамките на всеки цикъл изпращат към Q съобщението *msg*, което съдържа номера на процеса – 1 или 2. Консуматорът Q е недетерминиран – съдържа две алтернативи, чиито защити са команди за взаимодействие. Изпълнява се ($2 \times L$) пъти, което отговаря на общия брой съобщения, изпращани от P_1 и P_2.

В тялото на процесите P е предвидена задръжката Δ. Чрез нея се управлява интензитета на съобщенията, като най-високият интензитет е при задръжка $\Delta = 0$. Реализацията на командата за избор не би трябвало да зависи от интензитета на съобщенията.

На графа на системата от фиг. 2.14 е посочена и примерна регистрация на получените от Q съобщения. Регистрираната последователност трябва да има случаен характер, поради случайния характер на избора.

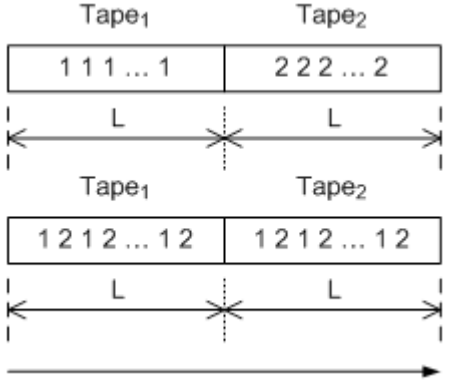


Фиг. 2.14. Паралелна система с недетерминизъм – два източника

Извършената с описаната постановка проверка на оператора *select* на *XC* за поддръжката на недетерминизъм дава резултата от фиг. 2.15. При най-тежкия случай ($\Delta = 0$) се проявява детерминирания характер на обработката на събитията, заложена в този оператор (горната половина на фигурата). В резултат, докато не завършат всички взаимодействия с P_1 не се преминава към взаимодействие с P_2. Предимството, давано на P_1 се обяснява с разположението на алтернативите в оператора *select*. Както даването на подобно „предимство", така и посоченият начин на отработване на събитията противоречат на изискванията.

Достатъчно е закъснението Δ да бъде ненулево (в конкретния експеримент е зададена стойност $\Delta = 1$, отговаряща на *10* ns) и обработката на събитията се нормализира – получаването на съобщенията започва да се редува (долната половина на фиг. 2.15). Детерминираният характер на оператора *select* обаче се потвърждава.

Необходимо е също да се проследят измененията, които



Фиг. 2.15. Изходна последователност при система с два източника

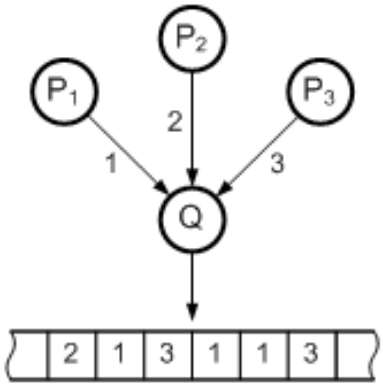
настъпват при увеличаване на броя на източниците на съобщения, а оттам и при увеличаване вероятността няколко събития да настъпят едновременно. Така се стига до опитната постановка с три източника, описвана от системата *CSP* уравнения

$$\{P_1||P_2||P_3||Q\},$$
$$P_1 = P_2 = P_3 = P = *\{Q!msg \rightarrow \Delta \rightarrow SKIP\}_L,$$
$$Q = * \begin{Bmatrix} P_1?x \rightarrow write(x) \rightarrow SKIP \\ \square\, P_2?x \rightarrow write(x) \rightarrow SKIP \\ \square P_3?x \rightarrow write(x) \rightarrow SKIP \end{Bmatrix}_{3\times L}. \quad (2.5)$$

Графът на тази паралелна тестова система от четири процеса съдържа и примерна регистрация на получените от *Q* съобщения (фиг. 2.16). Регистрираната последователност отново би трябвало да има случаен характер, поради случайния характер на избора.

На практика експериментът потвърждава предходните резултати – докато не спре изпращането на съобщенията от P_1, *Q* не може да премине към получаване на съобщенията от P_2. Съответно, докато не спре изпращането на съобщенията от P_2, *Q* не може да премине към получаване на съобщенията от P_3 (горната част на фиг. 2.17). Този резултат се запазва и при внасянето на минимална задръжка – до около *20* ns (*Δ = 2*). При следваща стъпка на задръжката от *30* ns (*Δ = 3*) започва да се редува получаването на съобщенията от P_1 и P_2 (средната част на фиг. 2.17). И едва при задръжка от *50* ns (*Δ = 3*) и нагоре започва редуването на получаването на съобщенията и от трите източника.

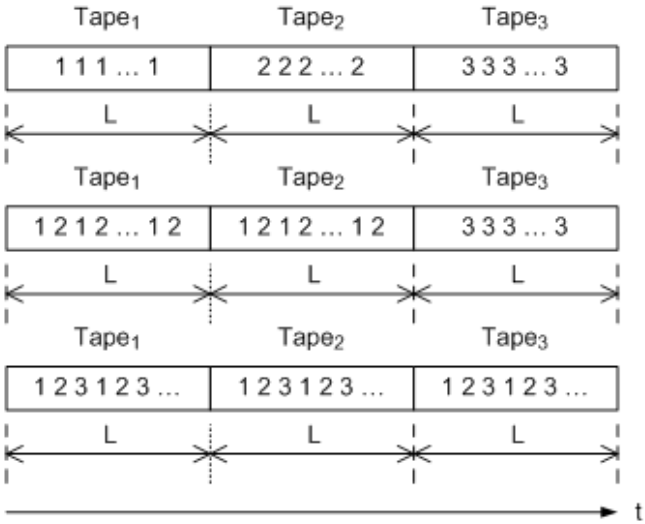
Разгледаният експеримент с оператора *select* е извършен с процесора *XS1-L1* и развойната среда *XDE* версия *11.11.0* на фирмата *XMOS*. Представлява интерес следващата проверка на реактивността на оператора при максимален брой източници на съобщения, който за *XS1-L1* е 7, а за *XS1-G4* е 31.



Фиг. 2.16. Паралелна система с недетерминизъм – три източника

Констатираните резултати определено са изненадващи. Те са най-вероятното обяснение защо в документацията на езика *XC* не се прави пряка съпоставка между оператора *select* и алтернативната команда, въпреки връзката между тях. Разбира се, резултатите в голяма степен се дължат на екстремната ситуация, целенасочено създадена и при двете тестови системи – максимално интензивен поток от входящи съобщения. Подобна ситуация е практически рядко срещана. Въпреки това, би трябвало да се предвидят начини за правилното ѝ отработване, което може да се постигне единствено чрез включването на недетерминизъм при обработката на събитията.

Включването на недетерминизъм би трябвало да използва в максимална степен съществуващата машинна поддръжка и



Фиг. 2.17. Изходна последователност при система с три източника

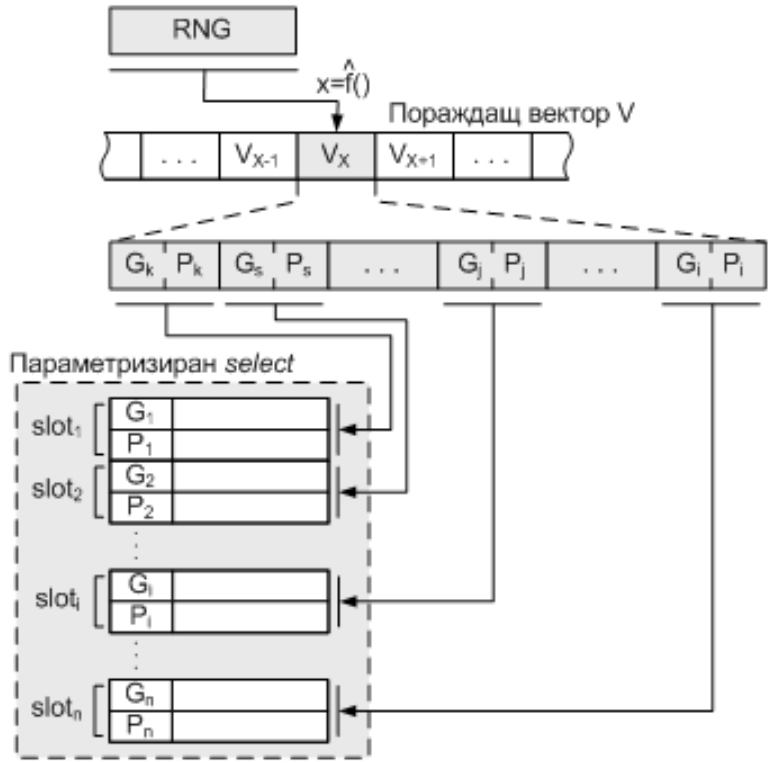
конструкциите на езика *XC*.

Предлаганата техника се основава на параметризирания вариант на оператора *select* на *XC* [49]. В този вариант операторът *select* се изпълнява като функция, чиито аргументи служат за параметризация на оператора. Идеята е преди обръщението към тази модификация на оператора *select* по случаен начин да се задава реда на алтернативите му.

Предварително се дефинира пораждащия масив *V*, който съдържа всички възможни уникални подредби на *n*-те алтернативи. Броят на тези *n* подредби на *n* елемента е частен случай на възможните вариации на *m* елемента от *n* при *m* = *n*

$$A_n^m = n(n-1)(n-2)\dots[n-(m-1)] = n(n-1)(n-2)\dots 1 = n! \quad (2.6)$$

При две алтернативи (*n* = *2*), възможните вариации са две, а при три алтернативи (*n* = *3*) – шест. Съответно всеки елемент V_x на пораждащия масив представлява вектор, съдържащ елементите на конкретната подредба. А всеки елемент на подредбата представлява структура от параметрите на алтернативата. Най-общо тази структура може да се отбележи като двойката <G_k, P_k>.



Фиг. 2.18. Включване на недетерминизъм при обработката на събитията

Преди обръщението към параметризирания оператор *select* по случаен закон се определя индекса *x* от пораждащия масив *V* за достъп до елемента му V_x. Съдържанието на елемента V_x формира параметрите за обръщение към оператора *select* и определя реда на алтернативите в него (фиг. 2.18).

За случайния избор може да се използва някой от разгледаните *RNG*. Много подходящ е генераторът с примитивна функция на средата от т. 2.2.

+++

Недетерминизмът не само е вътрешно присъщ за паралелните системи, той е и средство за ефективно обслужването на множеството асинхронни събития в този клас системи. Представеният експеримент нагледно показва как отсъствието на недетерминизъм води до намаляване реактивността на системата.

За преодоляване на този недостатък е препоръчително да се използва техника за вграждане на недетерминизъм при обслужването на събитията. Предложената в тази точка техника се основава на случаен избор и на параметризирания вариант на оператора *select*. За максимална ефективност на случайния избор се прилага машинната примитива *CRC32* на ядрото. В резултат се постига необходимата адекватност на оператора *select*, като реализация на алтернативната команда на *CSP*.

Техниката е подходяща за малък брой алтернативи (каквито са повечето практически случаи) поради пространствената ѝ сложност *O*(*n!*) и ограничения обем на вградената памет на фамилията *XS1*. При това, времевата ефективност на предложеното решение е константа - *O*(*c*). За пръв път е представена в [10].