Лекция 4

Задание условий движения инициатора

Предлагаемая модель описания процесса предполагает, что моменты сцепления инициатора с элементарными операторами определяют *сами* элементарные операторы. С этой целью введем в состав элементарного оператора, наряду с h_i^c, оператор h_i^y, определяющий условие, при выполнении которого инициатор покидает оператор h_i^c и сцепляется со следующим оператором h_{i+1}^c. Возможны следующие варианты задания такого условия:

а) указание момента времени сцепления инициатора с оператором h_{i+1}^c;

б) определение логического условия, при выполнении которого инициатор сцепляется с оператором h_{i+1}^c;

в) комбинированная форма, включающая варианты а) и б).

Таким образом: можно определить условный оператор, как:

$$h_i^y \in \left\{h_i^t, h_i^л, h_i^{t,л}\right\}, \text{ где}$$

h^y_i - оператор условия продвижения инициатора;

h^t_i - оператор временного условия (соответствует варианту а);

$h^л_i$ - оператор логического условия (соответствует варианту б);

$h^{t,л}_i$ - оператор комбинированного условия (соответствует варианту в).

Расширим понятие элементарного оператора, добавив к нему помимо оператора h^c_i оператор h^y_i. Таким образом, определим элементарный оператор h_i, как двойку:

$$h_i = \left\langle h_i^c, h_i^y \right\rangle.$$

Теперь можно определить понятие *алгоритмической модели процесса* (в дальнейшем АМП), в виде тройки:

$$\text{АМП} = \left\langle \{h_i\}_{i=1}^n, \beta, I \right\rangle,$$

где: $\{h_i\}_{i=1}^n$ - множество элементарных операторов;

β- линейный порядок на $\{h_i\}_{i=1}^n$;

I- инициатор.

Следует обратить внимание на то, что АМП содержит *один и только один* инициатор, т.е. каждому процессу соответствует *один* инициатор. В этом смысле инициатор является представителем процесса, при его потери либо отсутствии развитие процесса прекращается.

Линейную последовательность элементарных операторов назовем **треком** TR:

$$TR = \left\langle \{h_i\}_{i=1}^{n}, \beta \right\rangle$$

Тогда можно АМП определить также как двойку:

$$\text{АМП} = <\text{TR,I}>$$

Структура трека

Пусть задан некоторый трек TR. В реальных приложениях трек содержит достаточно много элементарных операторов, выполняющих одни и те же операции над аргументами. Операторы эквивалентны, если при одних и тех же значениях аргументов они вычисляют одинаковые результаты. Это свойство трека позволяет задать отношение эквивалентности на множестве $\{h_i\}_{i=1}^{n}$ элементарных операторов трека TR.

Назовем *структурой* свертку трека TR по отношению эквивалентности элементарных операторов.

Пример. Пусть задан некоторый трек TR (рисунок 4.1).

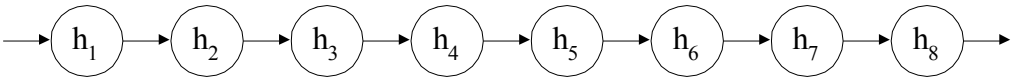


Рисунок 4.1. Пример трека

Пусть отношение эквивалентности элементарных операторов имеет вид:

$$\{(h_1, h_3), (h_2, h_5, h_6, h_8), (h_4, h_7)\}$$

Тогда структура имеет вид графа (рисунок 4.2).

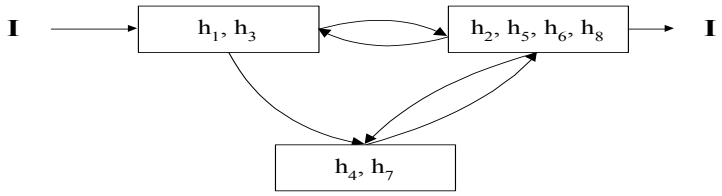


Рисунок 4.2. Свертка трека в структуру

Очевидно, если заданы трек и отношение эквивалентности операторов, то всегда возможно построение структуры. Однако обратное восстановление трека по структуре является неоднозначной операцией. Эта операция относится к классу операций развертки. С тем, чтобы операцию построения трека из структуры сделать однозначной, введем еще один тип элементарного оператора - *навигационный* элементарный оператор. Навигационный оператор определяется так же, как и элементарный оператор, однако в результате его выполнения определяется тот элементарный

оператор в структуре, который должен выполняться следующим. Выполнение навигационного оператора инициируется инициатором. Поскольку время на выполнение навигационного оператора, как и всех элементарных операторов, равно нулю, то использование его не сказывается на времени реализации процесса. В общем случае навигационный оператор должен следовать за каждым элементарным оператором в структуре.

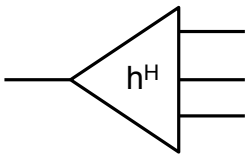


Рисунок 4.3 Обозначение навигационного оператора

Если навигационный оператор обозначить, как показано на рисунке 4.3, то структура из вышеописанного примера будет иметь вид (рисунок 4.4):

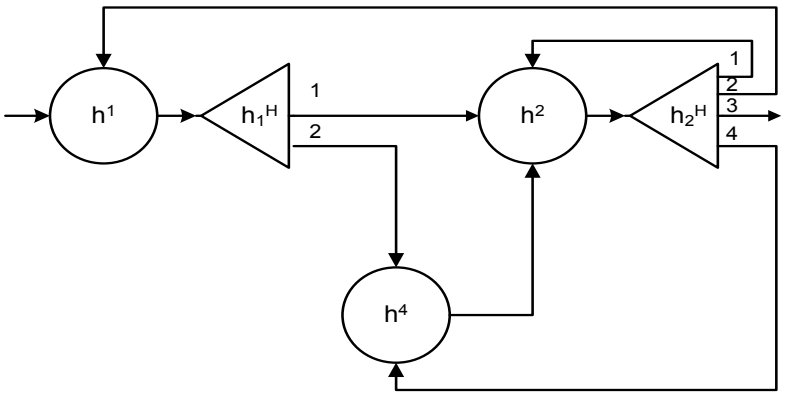


Рисунок 4.4. Вид структуры с навигационными операторами

Здесь операторы h_1, h_2, h_4 являются представителями своего класса эквивалентности. Как видно, после операторов h_1 и h_2 стоят навигационные операторы $h^{н}_1$ и $h^{н}_2$, в то время как после оператора h_4 нет необходимости в использовании навигационного оператора. Навигационный оператор используется также и для организации циклов (оператор $h^{н}_2$, выход 1).

Использование структуры по сравнению с треком позволяет значительно снизить размерность описания процесса. Однако необходимо иметь в виду, что процесс определен только в случае задания трека, а поэтому структура есть лишь способ более компактного описания трека, генерация самого трека остается необходимой операцией. На практике задание структуры с навигационными операторами для последующей генерации трека используется часто и повсеместно, где необходима генерация процесса.

С учетом вышеизложенного определение элементарного оператора в составе структуры можно представить в виде:

$$h=< h^c, h^y, h^n >.$$

Операторная схема

Элементарный оператор h оперирует с параметрами и изменяет состояние объекта. По отношению к оператору параметры могут быть входными, выходными и рабочими.

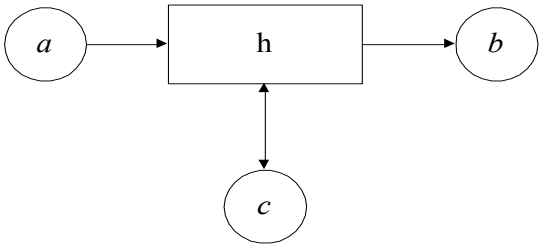


Рисунок 4.5.

где: *a* - входной параметр; *b* - выходной параметр; *c* - рабочий параметр.

Входной параметр означает его принадлежность к множеству *A*, выходной - к формированию состояния *s* , что говорит о его принадлежности к множеству O, рабочий - к тому и другому множеству одновременно.

Если на треке элементарных операторов указать используемые этими операторами параметры и их взаимосвязи, то получим операторно-параметрическую схему.

Пример такой схемы приведен на рисунке 4.6. Двойными линиями показан путь инициатора, а одинарными - отношение параметров к операторам. Такие схемы дают наглядную картину взаимодействия параметров в ходе реализации процесса.

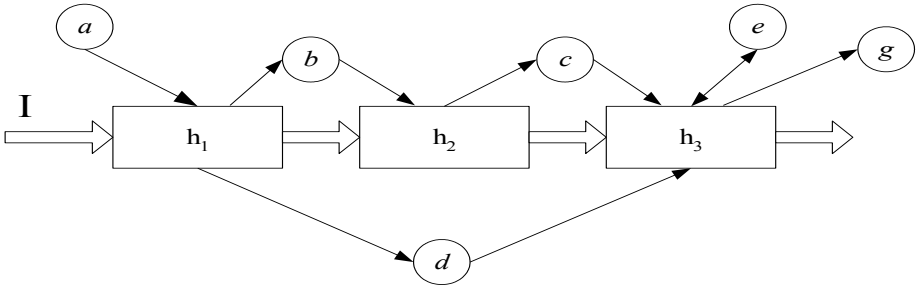


Рисунок 4.6. Операторно-параметрическая схема

Как видно из приведенного примера, операторно-параметрическая схема дает достаточно полное представление о способе описания процесса с использованием АМП.

Подобные процессы

Рассмотрим случай задания двух близких по описанию процессов Z_1 (трек A) и Z_2 (трек B), показанных на рисунке 4.7. И в том и в другом треке используются элементарные операторы h_1 и h_2, но они взаимодействуют с разными параметрами как входными, так и выходными. Было бы желательно найти способ объединения описаний таких процессов. Для решения поставленной задачи дополним определение инициатора, добавив к его фундаментальным свойствам возможность включать в себя параметры. Таким образом, инициатор наряду с фундаментальными свойствами приобретает некоторое "тело" в виде совокупности параметров. Параметры в этой совокупности должны быть упорядочены. Назовем эту совокупность параметров *локальной средой* процесса. Будем в дальнейшем считать, что "тело" инициатора представляет собой *ссылку на локальную среду*. Таким образом, движение инициатора по треку есть движение ссылки на локальную среду по треку, а сама локальная среда может быть неподвижна.

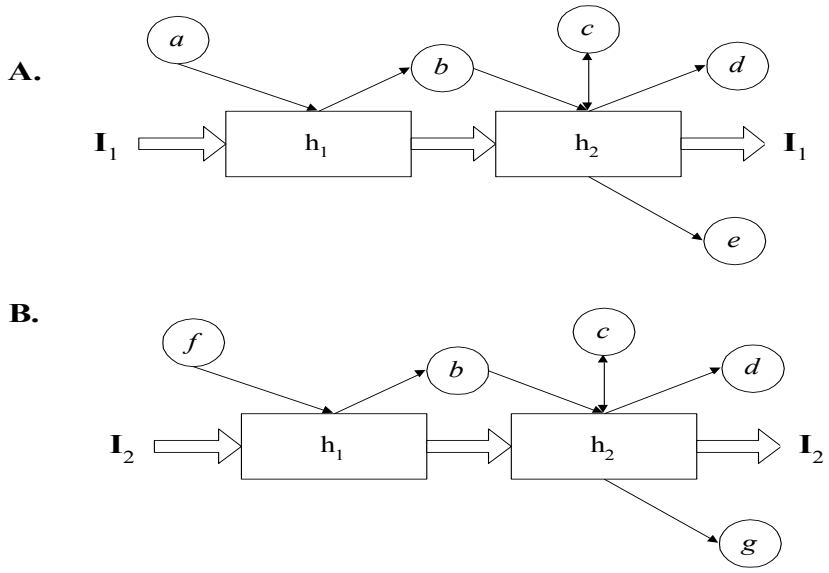


Рисунок 4.7. Подобные схемы

Тогда можно предложить следующую схему свертки описаний двух процессов (рисунок 7) в одно общее описание (рисунок 4.8):

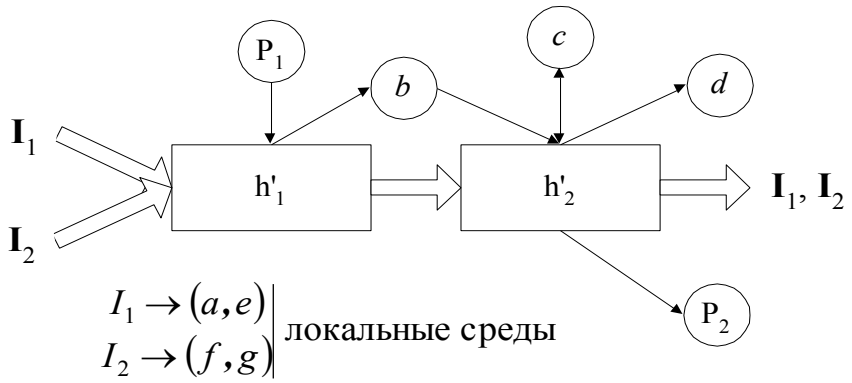


Рисунок 4.8. Свертка подобных схем

Поскольку элементарные операторы в такой схеме работают не только с явно заданными параметрами, но и с параметрами в локальных средах, то назовем такие элементарные операторы *объединенными* элементарными операторами.

Эти рассуждения могут быть распространены на случай **n** параллельно протекающих процессов. Процессы, сгенерированные треком или структурой, использующими объединенные элементарные операторы и локальные среды, называются *подобными*.

Таким образом, удается снизить размерность описания множества процессов, введя отношение подобия процессов. Для описания совокупности подобных процессов достаточно иметь одно объединенное описание трека или структуры и множество одинаково структурированных локальных сред, привязанных к инициаторам.

Ресурсы, конфликты на ресурсах

Процессы Zi в системе Q развиваются параллельно. Это значит, что они изменяют значения параметров системы в течение одного и того же интервала времени. Достаточно типичны ситуации, когда по логике функционирования системы накладываются ограничения на изменение некоторых параметров несколькими процессами одновременно в течение заданного либо обусловленного интервала времени. Это возможно лишь для пересекающихся объектов. Ограничения развития процесса, накладываемые другими процессами, назовем конфликтными ситуациями, или конфликтами.

Общую область параметров для пересекающихся процессов Ok и Om назовем ресурсом. Таким образом, ресурс $R_{k,m}$ определяется, как

$$\mathbf{R_{k,m} = O_k \cap O_m}$$

Конфликт возможен только при наличии ресурса. Таким образом, необходимо добиться согласования процессов в этой области. В основе способов разрешения конфликтов лежит утверждение об обязательном разделении доступа процессов к ресурсу во времени. Рассмотрим следующие способы разрешения конфликтных ситуаций.

А. Явное разделение по времени (синхронизация). При этом способе разрешения конфликта разнесение во времени производится явным указанием интервалов времени, определенных для каждого процесса. На рисунке 4.9 показан пример выделения таких интервалов для случая конфликта трех процессов.

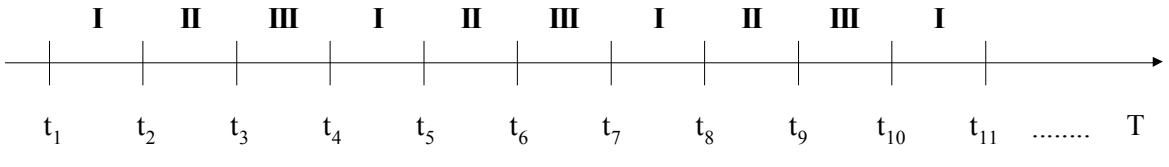
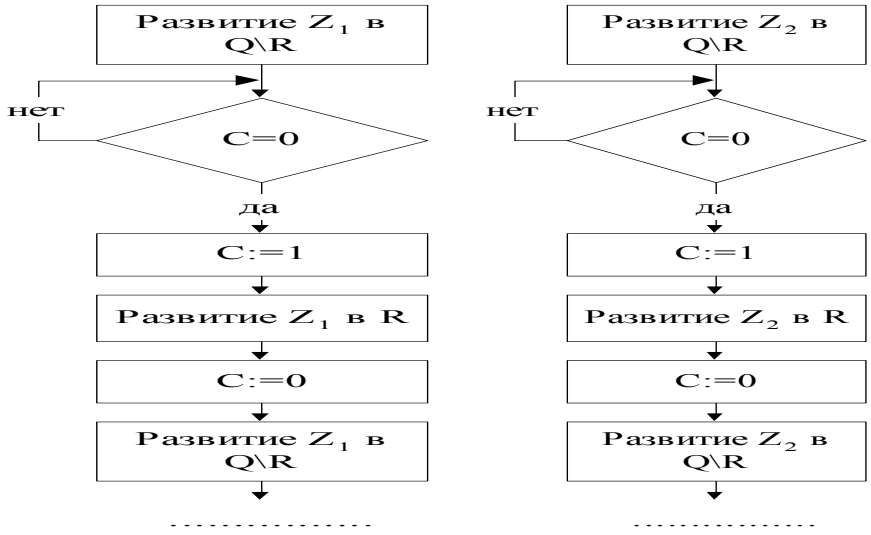


Рисунок 4.9. Разделение по времени

Б. Логический способ. В частности – использование семафоров. Если условие захвата ресурса не ограничивает время использования этого ресурса захватившим его процессом, то в этом случае удобно использовать семафоры. Семафор относится к логическим способам разрешения конфликтов. Семафор есть простая логическая переменная, однозначно соответствующая ресурсу. Значение семафора ‘0’ означает, что ресурс может быть захвачен процессом, значение семафора ‘1’ блокирует захват ресурса. На рисунке 4.10 показан пример использования семафора C при захвате ресурса R двумя процессами Z1 и Z2. Следует заметить, что в качества семафора может выступать любая логическая функция.

Рисунок 4.10. Использование семафоров



В. Алгоритмический способ. Наиболее общий способ управления процессами при захвате ресурсов состоит в использовании блоков - контроллеров. Использование этих блоков для разрешения конфликтов приведено ниже при определении понятия К-блока.

1.7. Блоки, типы блоков

На треке можно задать некоторое плотное разбиение элементарных операторов на подмножества. Это разбиение обычно выполняется с целью получения функционально однородных подмножеств операторов. Совокупность операторов, входящих в одно подмножество, назовем обобщенным оператором.

Пример разбиения приведен на рисунке 11. Здесь подмножество операторов h1, h2, h3 объединено в один обобщенный оператор H1, а h4, h5 - в обобщенный оператор H2. В результате получаем трек обобщенных операторов.

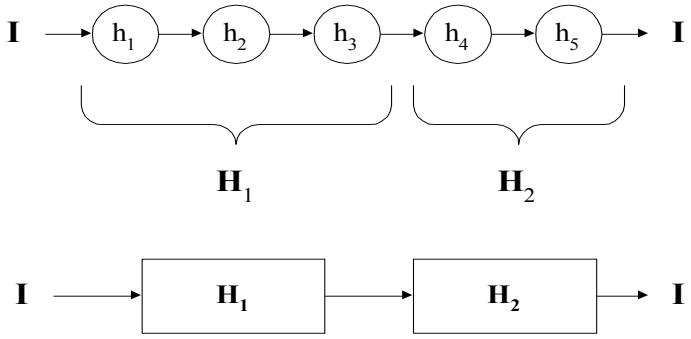


Рисунок 11. К понятию обобщенного оператора

Совокупность обобщенных операторов и связанных с ним параметров образует блок. Принципиально будем различать два типа блоков: агрегат и процессор.

Агрегат. На практике часто возникает необходимость описывать процесс функционирования некоторой машины по преобразованию значений параметров в соответствии с заданным циклическим алгоритмом. Такой процесс можно описать с помощью блока общего вида, в котором существует один инициатор, а трек циклически замкнут. Блок, в котором развивается один единственный циклический процесс, будем называть агрегатом, или А-блоком. Таким образом, агрегат содержит единственный инициатор и трек элементарных операторов, замкнутый внутри блока. Обмен между агрегатом и другими блоками возможен исключительно посредством параметров.

Процессор. Блок, предназначенный для генерации процессов, инициаторы которых являются внешними по отношению к блоку, назовем процессором, или П-блоком. Инициаторы, поступившие извне, сцепляются с блоком, порождая процессы, и затем покидают его. Поскольку процессор генерирует множество одновременно протекающих процессов, в нем используются исключительно объединенные элементарные операторы, а инициаторы должны содержать локальные среды. Таким образом, процессор представляет собой описание произвольной структуры, содержащей объединенные операторы. Процессы порождаются в этом блоке лишь при поступлении в него извне инициаторов, содержащих локальные среды. Из вышесказанного следует, что процессор порождает параллельно протекающие во времени подобные процессы.

Контроллер. Рассмотрим вновь блок типа агрегат. Как было показано выше, он не имеет возможности взаимодействовать с внешними инициаторами. С тем, чтобы снять это ограничение, введем над инициаторами операции пассивизации и активизации. Операция пассивизации переводит инициатор в класс обычных параметров. Операция активизации, наоборот, обычный параметр переводит в класс инициаторов. Если агрегат содержит операторы, выполняющие указанные операции, то такой агрегат назовем контроллером, или К-блоком. Контроллер, таким образом, представляет собой агрегат, выполняющий операции над внешними инициаторами в соответствии с собственным алгоритмом функционирования. Операции над инициаторами суть операции над

процессами. Таким образом, контроллер исполняет роль управляющего звена в некоторой блочной схеме.

На рисунке 12 показаны обозначения блоков, используемые в дальнейшем.

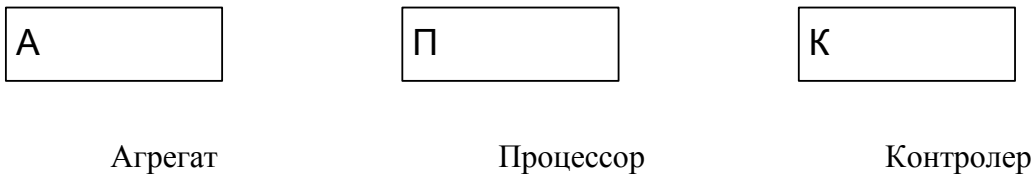


Рисунок 12. Обозначения блоков

Пример использования контроллера для разрешения конфликтной ситуации между процессами Z1 и Z2 приведен на рисунке 13.

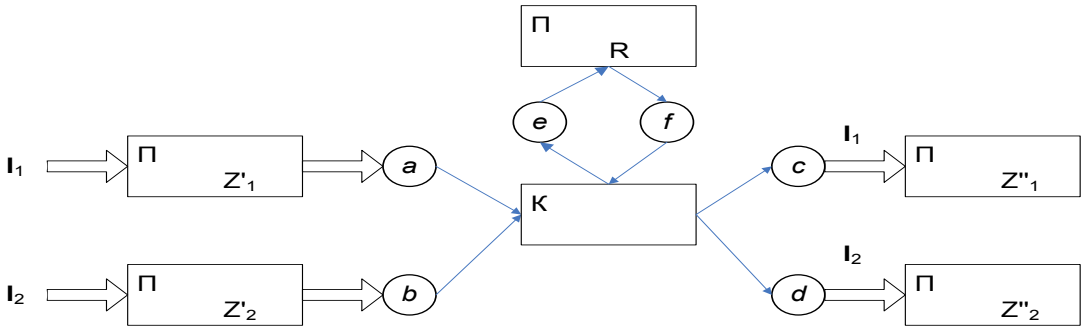


Рисунок 13. Применение К-блоков при разрешении конфликтов на R

Здесь первые П-блоки реализуют процессы Z1’ и Z2’ в пространстве Q\R; затем производится пассивизация инициаторов I1 и I2, они переводятся в параметры a и b соответственно. К-блок рассматривает ситуацию с входными параметрами в соответствие с собственным алгоритмом. Приняв решение о захвате ресурса каким-либо процессом, К-блок передает a или b в параметр e и активизирует его, отсылая в П-блок ресурса R. После завершения процесса в R выдается сигнал в параметре f, в ответ на который К-блок передает параметр - инициатор в c либо d и активизирует его, отсылая на продолжение процесса Z1’’ либо Z2’’ в соответствующие П-блоки. Использование К-блоков является наиболее универсальным способом управления множеством процессов при захвате ресурсов.

1.8. Схемы описаний функционирования систем

Агрегативная схема. Агрегативная схема описания функционирования системы предполагает использование только А-блоков (рисунок 14). Как показано выше, в такой схеме взаимодействие может осуществляться лишь через параметры. В результате формируется информационная сеть

агрегатов: агрегаты - вершины сети, дуги - информационные связи. Для агрегативных схем показано, что каждый А-блок в такой модели может представляться некоторым конечным автоматом. Если функционирование системы может быть описано совокупностью конечных автоматов, взаимодействующих между собой через множество входных и выходных параметров, то применение агрегативных схем представляется наиболее рациональным.

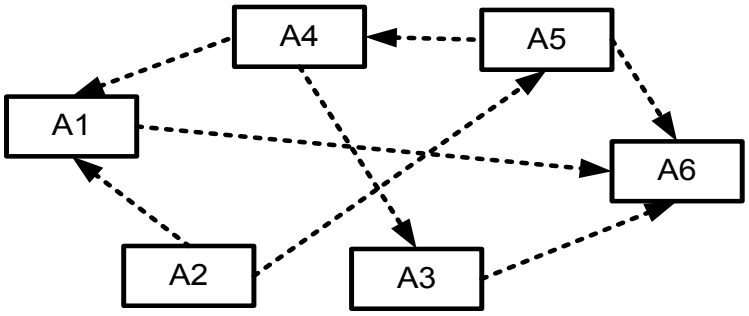


Рисунок 14. Пример агрегативной схемы

На рисунке 14 показаны 6 агрегатов, выполняющих функции сбора и обработки информации. Например, это могут быть какие-либо агентные структуры. Пунктирными линиями показано взаимодействие через параметры, относящиеся к тому или иному агрегату. Из схемы видно, что агрегат А2 сообщает информацию, агрегаты А1, А4, А3, А5 преобразуют полученную информацию, агрегат А6 собирает всю полученную информацию.

Процессная схема. Если в основе описания функционирования системы лежит использование П-блоков, то такое описание назовем процессным. Взаимодействие в процессных схемах осуществляется как через параметры, так и через локальные среды процессов. Процессный подход наиболее эффективен, когда имеем дело с множеством явно выраженных локальных процессов, например, при описании информационных, биологических, экономических, социальных и т.п. систем. Процессный подход реализован в языках Simula, GPSS и др.

Пример процессной схемы приведён на рисунке 15.

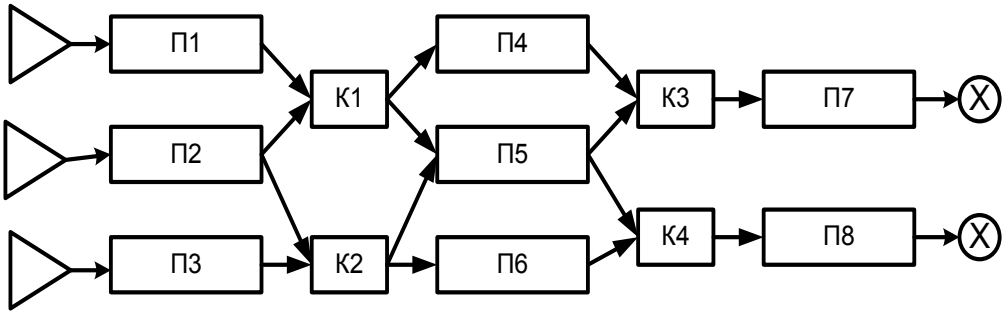


Рисунок 15. Пример процессной схемы

На рисунке 15 показаны процессоры и контроллеры, обеспечивающие управление потоками инициаторов и разрешение конфликтных ситуаций. Сплошными линиями показаны пути движения инициаторов. Как видно из схемы, инициаторы, порожденные генераторами, проходят все треки, заданные процессорами и уничтожаются к конце пути. В процессных схемах реализация всех процессов осуществляется только инициаторами, возникшими из генераторов. Контроллеры лишь меняют направление и условия продвижения инициаторов.

Потоковая схема. Потоковая схема возникает из агрегативной, когда среди связей агрегатов появляются связи с управляющей информацией. Когда эта информация поступает в агрегат, она производит инициацию процессов внутри агрегата. Если на графе информационных связей агрегативной схемы выделить лишь связи с управляющей информацией, то получим потоковую сеть агрегатов. Управляющая информация в этом случае может рассматриваться, как инициаторы сетевого уровня.

Таким образом, в потоковых схемах существуют два вида инициаторов: сетевые инициаторы, инициирующие запуск агрегата, и блочные инициаторы, реализующие процесс функционирования блока. Пример потоковой схемы приведен на рисунке 16.

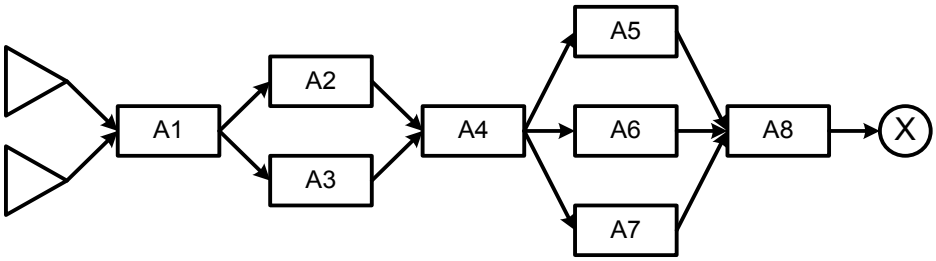


Рисунок 16. Пример потоковой схемы

На рисунке стрелками показаны пути движения сетевых инициаторов. Блочные (внутренние инициаторы) находятся внутри агрегатов. Например, сетевой инициатор, поступающий на агрегат А4 от агрегата А2, вызывает выполнение некоторого процесса в А4. Сам инициатор после этого может быть уничтожен или перемещен блоком А4 дальше по сети.

К потоковым схемам относятся сети Наура, Е-сети, сети Петри. Наибольшее распространение среди потоковых схем получили сети массового обслуживания (СеМО). Эти сети возникают из процессных схем, когда процессоры очень просты, либо отсутствуют вообще. Тогда основным элементом сети становятся контроллеры. Объединение контроллера с простым процессором или агрегатом формируют объект, называемый системой массового обслуживания (СМО). Таким образом, возникает сеть из СМО. Согласно определению потоковой схемы, по этой сети перемещаются инициаторы сетевого уровня, которые в теории СеМО называются сообщениями или требованиями.