Труды VI-й Международной конференции "Параллельные вычисления и задачи управления" (PACO'2012). Том 2. - М., 2012. - с. 244-250.

© 2012 г. В.С. ВЫХОВАНЕЦ, д-р техн. наук
А.В. ЯЦУТКО
(Институт проблем управления им. В.А. Трапезникова РАН)

РЕАЛИЗАЦИЯ ЕСТЕСТВЕННОГО ПАРАЛЛЕЛИЗМА ПРОЦЕССОВ НА ОСНОВЕ МОДЕЛЕЙ С СОВМЕЩЕННЫМИ СЕТЯМИ УПРАВЛЕНИЯ И ДАННЫХ

Рассматривается новый подход в области динамического управления процессами, основанный на использовании графической нотации с совмещёнными сетями управления и данных. Благодаря явному заданию потока данных появляется возможность реализации естественного распараллеливания и эффективное выполнения параллельных процессов.

THE IMPLEMENTATION OF NATURAL PROCESS PARALLELISM BASED ON THE MODEL WITH THE COMBINED CONTROL AND DATA NETWORKS / V.S. Vykhovanets, A.V. Yatsutko (V.A. Trapeznikov Institute of Control Sciences, Profsoyuznaya 65, Moscow 117342, Russia, E-mail: valery@vykhovanets.ru). A new approach in dynamic process control, based on the use of graphical notation with combined control and data networks. Due to the explicit instructions of the data-flow it is possible to implement a natural and efficient parallel execution of parallel processes.

1. Введение

Современные системы управления технологическими и производственными процессами представляют процесс в виде сети управления, связывающей между собой подпроцессы – узлы сети, дугами передачи управления от одного подпроцесса к другому,), задающими частичный порядок выполнения подчинённых процессов [1].

У любого реального процесса многие подпроцессы могут быть выполнены параллельно, причем такую ситуацию сложно описать на этапе создания графической модели. Последнее связано с тем, что параллельное или последовательное выполнение подпроцессов зависит не только от указываемых явно статических связей по управлению, но и от динамически возникающей готовности входных и выходных данных или их частей. Существующие технологии управления распараллеливанием не учитывают динамический характер частичного порядка, возникающего между подпроцессами. Такой динамически изменяющийся частичный порядок подпроцессов некоторого процесса будем называть естественным параллелизмом задачи, решаемой этим процессом.

Для случаев, когда процесс содержит подпроцессы, которые могут быть выполнены параллельно, предусмотрены специальные узлы («логическое «И», многоэкземплярные циклы параллельного типа, развёрнутые подпроцессы с параллельными блоками, шлюзы «ИЛИ» с неэксклюзивным условием, и т.п.), позволяющие явно задавать параллельно выполняемые подпроцессы [2].

Данные между подпроцессами, как правило, передаются неявно – через атрибуты процессов (по связям управления) или через специальные объекты синхронизации, реализуемые общей шиной данных или коммутационной сетью. Такая организация работы

с данными исторически наследуется с самых первых методологий анализа предметных областей, это обусловлено тем, что выделение потока данных сложно реализовать и методически, и технически [3]. Отсутствие явно задаваемого потока данных приводит к невозможности автоматического создания исполняемых моделей процессов, а также затрудняют актуализацию уже выполняющихся процессов.

Отсюда актуальным видится создание технологии распараллеливания процессов, которая позволит реализовать естественный параллелизм решаемой задачи. Такая технология должна включать в себя не только новое графическое представление взаимодействующих подпроцессов, но и средства реализации естественно возникающей и динамически изменяющейся готовности подпроцессов к выполнению.

2. Совмещённая сеть управления и данных

Появление проблем, свойственных современным средствам автоматизированного управления технологическими и производственными процессами, следует связать с недостаточностью данных, отражаемых на аналитических диаграммах процессов, их нечёткой пространственно-временной структурированностью и несогласованностью типов. Для преодоления этих проблем предлагается использовать аналитические модели процессов, содержащие не только точно специфицируемые связи процессов по управлению, но и точно специфицируемые источники данных и пути их передачи [4].

2.1. Графическая нотация

Для решения поставленной задачи предлагается использовать графические модели процессов, содержащие не только точно специфицируемые связи подпроцессов по управлению, но и точно специфицируемые источники данных и пути их передачи. Иными словами предлагается объединить выразительные возможности двух моделей – моделей для описания потоков работ и моделей для описания потоков данных. Совмещённая сеть даёт возможность описать синхронизацию подпроцессов как по управлению, так и по данным.

Через каждый узел сети (показан в виде прямоугольника) проходит поток управления и поток данных (рис. 1). Поток управления, входящий в узел, называется активатором (ромб). Получение активации означает начало работы узла. Выход потока управления из узла называется событием (квадрат). Событие, созданное узлом, может активировать один или несколько других узлов. Поток данных передаёт узлу структурированные входные данные (треугольник). После того, как узел закончил работу, в поток данных поступают структурированные выходные данные (окружность), которые могут передаваться следующему узлу в качестве входных. При срабатывании одного или нескольких активаторов узел активируется, считываются входные данные и выполняются действия, привязанные к данному узлу. Затем формируются одно или несколько событий и выходные данные.

При активации одного или нескольких активаторов составного узла в соответствии с заданными связями активизируются внутренние узлы. После окончания функционирования внутренних узлов их события по внутренним связям активизируют другие внутренние узлы. Функционирование составного узла завершается, когда все внутренние узлы перестают активироваться и закончили своё функционирование. Аналогично связям активации происходит передача данных между внутренними узлами и формирование выходных данных составного узла.

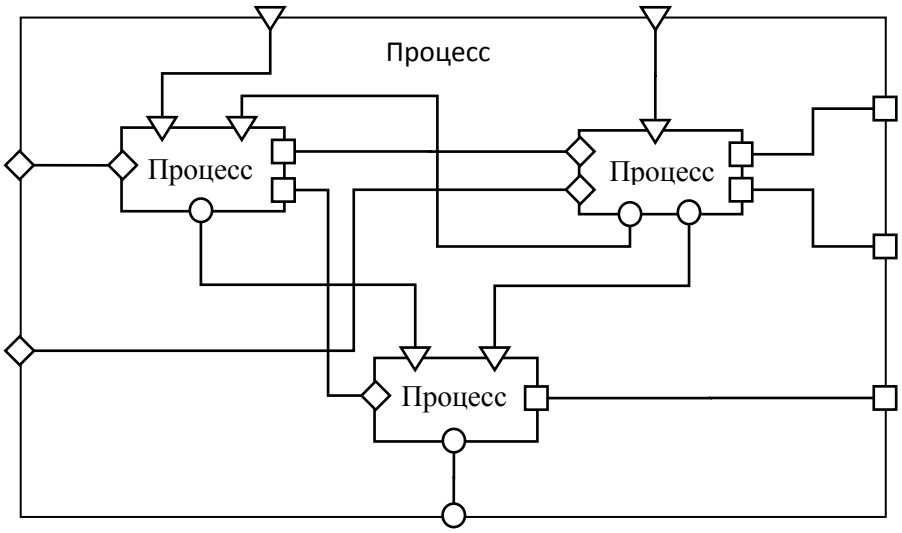

Рис. 1. Совмещённая сеть управления и данных

Часть узлов, которые не имеют своей совмещенной сети, называются базовыми. К базовому узлу привязывается последовательность действий (операций), заданная на языке некоторого интерпретатора: процессора, виртуальной машины, параллельного потока, какого-либо исполнительного устройства. В свою очередь составной узел строится из других узлов, в том числе и составных, а его вычислительная семантика однозначно определяется его совмещенной сетью.

В связи с тем, что составной узел содержит явное указание на принимаемые и передаваемые данные для всех внутренних узлов, то распараллеливанию подлежат все активированные внутренние узлы. Следует обратить внимание на то, что именно здесь проявляется динамический характер распараллеливания, заключающийся в том, что в каждый конкретный момент времени и в зависимости от предыстории развития процесса принимается решение о том, какие узлы подлежат параллельному выполнению, а какие нет. Например, может оказаться так, что при одном и том же описании узла в один момент времени параллельному выполнению подлежит одно подмножество внутренних узлов, а в другой момент времени – другое.

Например, при выполнении составного узла описания активированных внутренних узлов могут записываться в специальную таблицу. Затем один или несколько интерпретаторов выбирают из этой таблицы для выполнения самые приоритетные узлы, которые являются базовыми, имеют большую глубину вложенности или все составные узлы которого оказались исполненными.

2.2. Дифференциация входов и интеграция выходов

Особенностью интерпретации совмещённых моделей является дифференциация активаторов и входных данных, и интеграция событий и выходных данных [5]. Процесс дифференциации активаторов составного узла состоит в фиксации фактов активации таким образом, что воздействие таких активаторов на связанные с ним внутренние узлы осуществляется однократно и только в момент активации. Процесс интеграции событий составного узла состоит в накоплении событий, поступающих от внутренних узлов за все время их функционирования. Если событие составного узла было возбуждено хотя бы один раз, то это является основанием для принятия решения о передаче их числа

внешнему узлу, включающего текущий узел в качестве составного. При этом число возбуждений события интерпретируется как сила активация узла или узлов, активаторы которых соединены с этим событием.

Аналогичным образом процесс дифференциации входных данных заключается в их фиксации в момент активации узла. В свою очередь, интеграция выходных данных осуществляется объединением данных на выходе данных. Интегрирование выходных данных осуществляется путём поимённого объединения частей данных, получаемых после окончания функционирования каждого внутреннего узла, соединённого с выходом данных. Для такого объединения данные представляются именованными списками значений вида (Имя, Значение 1, Значение 2, …), где любым из значений может быть другой именованный список. Такую структуру данных удобно представлять деревом, у которого корнем является выходной порт, а листья – значения данных или другие деревья (рис. 8). Например, для данных на рис. 2 именованный список представляется так:

(Выход 1, (Имя 1, …), (Имя 2, Значение 1, (Имя 3, …)), Значение 2, Значение 3).

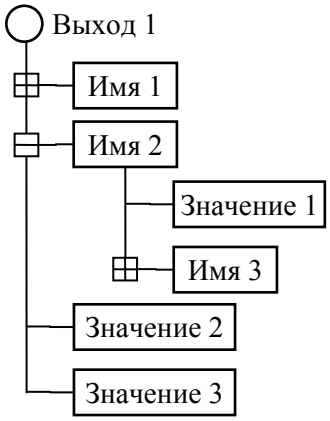

Рис. 2. Древовидная структура данных

Следует обратить внимание на то, что рассматриваемая древовидная структура может использоваться для спецификации интерфейсов входных и выходных данных узлов. В этом случае при конструировании узла задаются структуры данных, минимально необходимые (непосредственно используемые) при его функционировании. Если во время активации такого узла на его вход поступят данные, не содержащие их минимальный объем, может фиксироваться ошибка передачи данных, что сигнализирует о неправильной работе узла – поставщика этих данных. В свою очередь во время конструирования узлов могут быть выданы предупреждающие сообщения о том, что выполненная трассировка данных не является корректной, так как выходной порт узла – поставщика данных не содержит данных, необходимых для правильного функционирования узла – их получателя.

2.3. Выполнение совмещённых моделей

Базовые узлы исполняются непосредственно тем исполнителем, который определён во время их конструирования, так как их вычислительная семантика задаётся явно. Исполнение составного узла – это исполнение всех его внутренних узлов. После дифференциации активаторов и входных данных исполняемого составного узла выполняется активация его внутренних узлов и пересылка входных данных в соответствии с внутренней трассировкой. Те узлы, которые активированы, передаются на исполнение. По-

сле окончания функционирования каждого из активированных узлов процесс исполнения повторяется до тех пор, пока имеются узлы, активированные на предыдущем шаге. При этом всякий раз, когда внутренний узел закончит функционирование, выполняется интеграция событий и выходных данных.

Для составных узлов имеется три подхода к реализации их исполнения (рис. 3) – рекурсивный интерпретатор, циклический интерпретатор и компиляция [6].

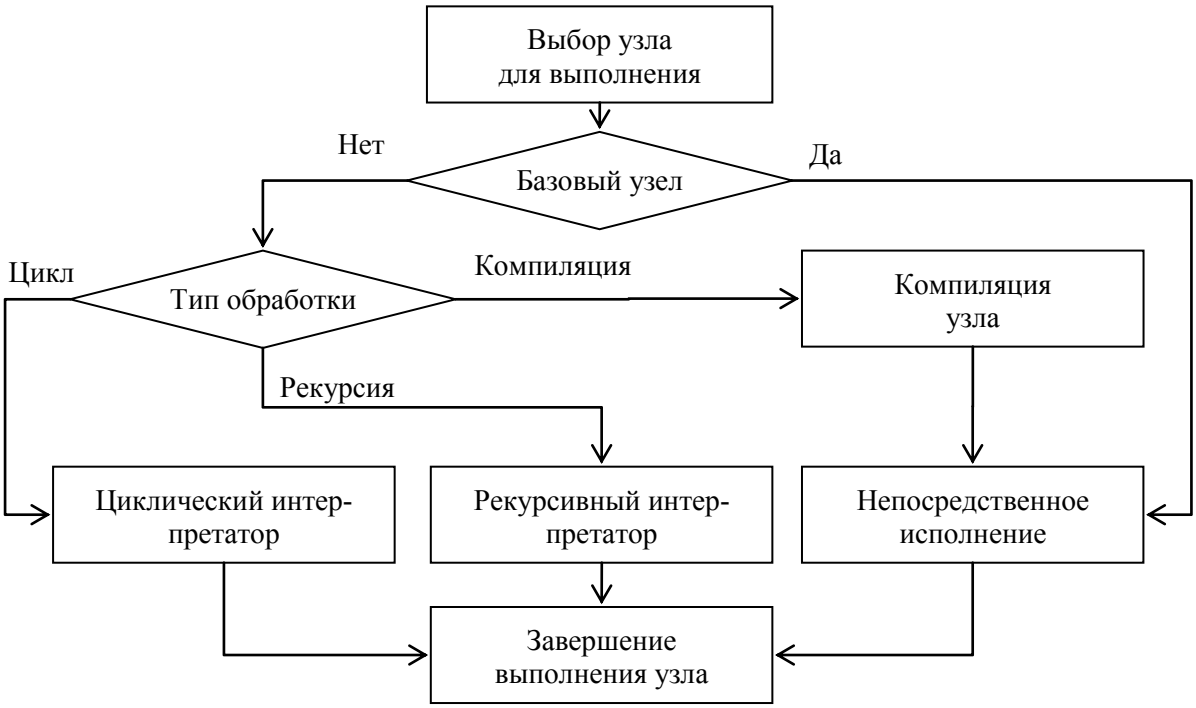

Рис. 3. Выполнение узла совмещённой модели

При рекурсивной интерпретации составного узла после определения активированных внутренних узлов выполняется непосредственное исполнение активированных базовых узлов и рекурсивный вызов того же интерпретатора для каждого активированного составного узла. Рекурсивную интерпретацию следует применять для быстро исполняемых составных узлов с небольшим уровнем вложенности, так как при рекурсивной обработке на все время исполнения составного узла резервируется полный объем вычислительных ресурсов, необходимый для исполнения всех вложенных узлов.

При циклической интерпретации составного узла описания активированных внутренних узлов записываются в специальную таблицу. Затем один или несколько циклических интерпретаторов выбирают из этой таблицы для исполнения самые приоритетные узлы. Приоритет отдаётся тем узлам, которые являются базовыми, имеют большую глубину вложенности или все составные узлы которого оказались исполненными. Циклическую интерпретацию следует применять при исполнении составных узлов с большим временем функционирования, вызванным, как правило, ожиданием реакции на некоторые внешние события.

Третий способ реализации исполнения составных узлов – их компиляция. Компиляция составных узлов основана на том, что вычислительная семантика как базовых, так и составных узлов полностью определяется их описаниями. Это позволяет автоматически породить программу на некотором языке программирования, которая после компиляции реализует исполнение откомпилированных узлов. Компиляцию следует применять для

простых и часто используемых составных узлов, так как их актуализация требует больших вычислительных ресурсов.

2.4. Динамическое распараллеливание

Нотации многих известных методологий моделирования процессов включают в себя специальные элементы для явной (статической) организации параллельного их выполнения. Однако у явного управления распараллеливанием процессов имеются существенные недостатки. Во-первых, качественно снижается наглядность модели, которая особенно важна на этапах её анализа и верификации. Во-вторых, для параллельного выполнения теряется большая часть запускаемых процессов, распараллеливание которых не может быть в явном виде указано на диаграммах по многим причинам. В-третьих, разработка диаграммы с параллельным выполнением процессов представляет собой, как правило, сложный, высококвалифицированный и непроизводительный труд.

Моделирование процессов в формализме совмещённых сетей управления и данных позволяет реализовать метод динамического (неявного) управления распараллеливанием, основанный на использовании естественного параллелизма процессов. В связи с тем, что составной узел содержит явное указание на принимаемые и передаваемые данные между внутренними узлами, то при его исполнении распараллеливанию подлежит исполнение всех активированных узлов. Следует обратить внимание на то, что именно здесь проявляется динамический характер распараллеливания, заключающийся в том, что в каждый конкретный момент времени и в зависимости от предыстории развития моделируемого процесса принимается решение о том, какие узлы подлежат параллельному исполнению, а какие нет [7]. Может оказаться так, что при одном и том же описании узла в один момент времени параллельному исполнению подлежит одно подмножество внутренних узлов, а в другой момент времени – другое.

Наиболее эффективно динамическое распараллеливание проявляется при циклической интерпретации узлов, где выбор узла для исполнения происходит среди множества активированных узлов, в том числе и принадлежащих различным моделируемым процессам. В этом случае реализуется динамическая балансировка загрузки интерпретаторов (процессоров), проявляющаяся в том, что в зависимости от траекторий развития моделируемых процессов более нуждающийся в обслуживании процесс получает больше вычислительных ресурсов, а менее нуждающийся – меньше.

3. Заключение

Применение методологии анализа и моделирования процессов на основе совмещённых сетей управления и данных позволяет преодолеть накопившиеся проблемы в области промышленных систем управления процессами.

Описанное выше автоматическое динамическое распараллеливание процессов позволяет добиться высокой эффективности вычислений, отличается возможностью масштабирования вычислений на различное число параллельно работающих процессоров (виртуальных машин, параллельных потоков). Более того, процессоры (виртуальные машины, параллельные потоки) автоматически подстраиваются под траектории развития запущенных процессов, реализуют динамическую балансировку своей загрузки.

При этом предлагаемая графическая нотация содержит небольшое число исходных элементов и простые правила их интерпретации, является такой же понятной аналитикам, как и другие зарекомендовавшие себя нотации. Немаловажным также является возможность актуализации запущенных моделей процессов при их естественном изме-

нении, что позволяет существенно улучшить качество управления современным динамически изменяющимся предприятием.

Использование структурированных входных и выходных данных процессов создает новое выразительное средство, облегчающее верификацию создаваемых моделей и отсутствующее у современных средств моделирования, а именно – спецификация интерфейсов входов и выходов данных, которая позволяет обнаружить и устранить ошибки трассировки данных на этапе моделирования.

Научная новизна предлагаемой технологии заключается в том, совместная и раздельная трассировка данных и управления между узлами сети предоставляет гибкий механизм синхронизации процессов, как по данным, так и по управлению, позволяет реализовать их естественное распараллеливание и эффективное выполнение.

СПИСОК ЛИТЕРАТУРЫ

1. *Davis R.* Business Process Modeling with ARIS: A Practical Guide. New York: Springer-Verlag, 2005. 531 p.
2. *Smith H., Fingar P.* Business Process Management (BPM): The Third Wave. Meghan-Kiffer Press, 2002. 312 p.
3. *Марка Д.* Методология структурного анализа и проектирования SADT. М.: Мета-Технология, 1993. 247 с.
4. *Яцутко А. В.* Моделирование бизнес–процессов крупномасштабного производства // Труды III Международной конференции «Управление развитием крупномасштабных систем». М.: ИПУ РАН, 2009. Том II. С. 301–303.
5. *Яцутко А. В.* Актуализация моделей бизнес-процессов в совмещённых сетях управления и данных // Сборник трудов Второй российской конференции с международным участием «Технические и программные средства систем управления, контроля и измерения». М.: ИПУ РАН, 2010. С. 1268-1272.
6. *Яцутко А. В.* Параллельные вычисления в методах теории управления на основе совмещенных сетей данных и управления // Доклады 5-ой Международной конференции «Параллельные вычисления и задачи управления». М: ИПУ РАН, 2010. С. 519-528.
7. *Яцутко А.В.* Синхронизация данных в автоматных сетях с раздельной трассировкой управления и данных // Сборник трудов Третьей российской конференции с международным участием «Технические и программные средства систем управления, контроля и измерения». М.: ИПУ РАН, 2012. С. 2060-2070.