УДК 658.512

М.В. Жуковская

ПАРАЛЛЕЛЬНЫЙ ЛОГИЧЕСКИЙ ВЫВОД В ПРОЦЕССОРНЫХ СТРУКТУРАХ.

В продукционных системах, ориентированных на языки типа Пролог, знания о предметной области должны быть представлены в виде импликативных правил (т.н. дизъюнктов Хорна):

$(P_1 \& P_2 \& ... \& P_n) \Rightarrow Q$,

где P_i - условия вывода, а Q - заключения.

Это связано с тем, что режим возвратов, основной инструмент поиска доказательства в последовательных продукционных системах, ориентирован прежде всего на обработку конъюнктивно связанных подцелей. Несмотря на то, что синтаксис некоторых версий языка Пролог допускает соединение подцелей с помощью дизъюнкции, это чисто внешнее соглашение, сделанное для улучшения читабельности программ. Интерпретатор Пролога преобразует такие конструкции в несколько правил с одинаковым консеквентом, каждое из которых содержит только операции конъюнкции [1].

Процесс доказательства в продукционных системах включает поиск применимых в текущем состоянии продукций и активизацию одной или нескольких таких продукций. Очевидно, что основной продукционный цикл может быть распараллелен. ИЛИ-параллелизм, используемый практически во всех параллельных реализациях продукционных систем, предполагает параллельное выполнение продукций, консеквенты которых сопоставимы с выполняемой целью. Использование другого вида параллелизма, присущего продукционной модели представления знаний, а именно, И-параллельности (параллельного выполнения подцелей, находящихся в целевом предложении), связано с серьезными проблемами. Основной из них является необходимость согласования результатов доказательства конъюнктивно связанных подцелей, имеющих общие переменные. Если две или более подцелей имеют общую (разделенную) переменную, то система вывода должна гарантировать, что при параллельном разрешении этих подцелей данная переменная будет связана одним и тем же значением. Двумя крайними решениями данной проблемы являются, с одной стороны, переход к последовательному перебору для подцелей с разделенными переменными, и, с другой стороны, запрещение вхождения в продукции разделенных переменных.

Параллельный поиск доказательства в принципе позволяет расширить синтаксис продукций путем введения эффективной обработки дизъюнктивно связанных подцелей:

$(P_1 \varpi_1 P_2 \varpi_2 ... \varpi_{n-1} P) \Rightarrow Q$,

где P_i - условия вывода, Q - заключения, а ϖ_i - связки, $\varpi_i \in (\&, \vee)$.

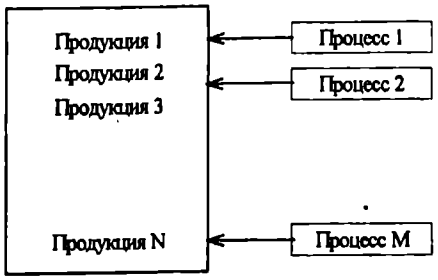


Рис. 1. Поиск доказательства в многопроцессорной системе с общей памятью.

Наиболее часто распараллеливание работы продукционных систем производят путем распределения процесса доказательства между несколькими процессорами. Естественной

архитектурой при таком подходе к обработке продукций является многопроцессорная система с общей памятью. К глобальной базе данных в этом случае обращается множество процессов (рис. 1), причем каждый процессор ведет один или несколько таких процессов, т.е. выполняет процедуру унификации и генерацию новых подцелей (будем называть данную схему параллельного доказательства "традиционной"). Доказательство целевого выражения начинает один процесс, который впоследствии делится на подпроцессы, причем при реализации И-параллелизма для конъюнктивно связанных подцелей необходимо предусмотреть механизм слияния результатов их доказательства. Таким образом, функциональные различия связок & и $\vee$ учитываются уже на этапе деления процессов. Преимуществом ведения доказательства с помощью параллельных взаимосвязанных процессов можно назвать то, что сразу после прекращения существования процессов системе обработки знаний становится известен результат доказательства, т.е. формирование ответа не отделяется от поиска доказательства. Очевидно, что это оплачивается большими накладными расходами на согласование процессов.

Многопроцессорная система с общей памятью не является единственно возможной архитектурой для системы обработки знаний (СОЗ), представленных в виде продукций. Уже разработаны и исследованы специализированные параллельные машины логического вывода, в том числе с модульным принципом хранения продукций. Асинхронность и естественный параллелизм продукционного цикла делает перспективным использование процессорных структур для построения СОЗ.

Под процессорной структурой понимают сеть, образованную вычислительными модулями, т.е. процессорами, снабженными локальной памятью. Разместив глобальную базу знаний в вершинах такой сети, можно расширить последовательный поиск применимых продукций, характерный для систем с общей памятью, до поиска по связной структуре, и, кроме этого, появляется возможность одновременного построения всех (нескольких) путей графа вывода.

Наиболее подходящей формой представления знаний для обработки в процессорных структурах принято считать различные графовые формы (сети). Для продукционных моделей в качестве базового представления знаний можно использовать графы типа И/ИЛИ [2]. Единицей информации, хранящейся в каждом процессорном модуле, в этом случае можно принять фрагмент сети, содержащей вершину (лексему) и ее связи с соседними вершинами. На множестве всех лексем M базы знаний определяется разбиение с учетом функциональной специализации:

$M = \{\{L\},\{R\},\{F\},\{G\},\{K\},\{D\}\}$,

где L-множество всех отрицательных литералов дизъюнктов, включая целевые; R-множество положительных литералов дизъюнктов; F-множество фактов (аксиом); G-положительные литералы целевых дизъюнктов; K,D-соответственно операции конъюнкции и дизъюнкции. Таким образом, каждый процессорный модуль будет осуществлять обработку информации строго определенного вида, в зависимости от типа хранящейся в нем вершины.

Процесс доказательства при данном представлении продукций сопровождается наращиванием графа целевой продукции за счет присоединения к нему графов правил и фактов (рис. 2). Правила и факты применяются к целевому выражению до тех пор, пока не будет получен полный граф поиска решения или процесс вывода не будет остановлен. Система успешно доказывает цель, если удается построить дерево типа И/ИЛИ (граф решения), все ветви которого заканчиваются в вершинах-фактах и все подстановки, помечающие дуги соответствия, согласованы.

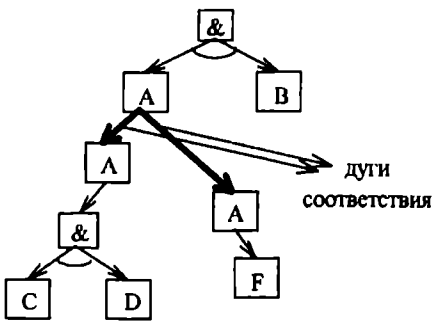


Рис.2. Применение двух правил: C&D⇒A и F⇒A к целевому выражению A&B

Процесс поиска доказательства в процессорной структуре, т.е. построение всех ветвей графа вывода, связанных как &, так и $\vee$, можно производить без анализа вида связок. Асинхронное

построение всех ветвей графа вывода без их взаимной координации с одной стороны, исключит задержки при поиске доказательства, а с другой стороны, приведет к построению полного графа решения, который является объединением всех возможных графов-кандидатов (решений). Даже после его полного построения нельзя будет судить о результате доказательства. В связи с этим встает проблема извлечения решения из найденного графа вывода, т.е. выявления графов-кандидатов и проверки их на согласованность.

Отличие данной схемы доказательства в процессорной структуре от традиционных методов показано на рис.3.

Рис.3. Доказательство целевого выражения при последовательной схеме (а), при традиционной параллельной схеме (б) и в процессорной структуре (в).

Извлечение ответа из полного графа вывода решает задачу, аналогичную поддержанию координации порожденных подпроцессов при традиционном подходе к параллельному логическому выводу, основанному на ведении множества параллельно работающих и взаимодействующих друг с другом процессов. Процесс последовательного извлечения ответа из полного графа решения, заключающийся в просмотре дерева в прямом направлении - от корня, и выявление графов кандидатов, является трудоемким. В сущности, он повторяет стратегию продукционного вывода с возвращениями, и отличается от него только тем, что поиска продукций не происходит, поскольку структура полного графа уже была определена. Для процессорных структур представляется целесообразным использование другого способа выявления графов-кандидатов, а именно, просмотр дерева решения с листьев, что позволяет отказаться от режима возвратов и использовать функциональные отличия вершин $v_j \in \{K\}$, $v_j \in \{D\}$, самым естественным образом.

Распределенное извлечение ответа предполагает перемещение множества подстановок, характеризующее конкретный граф-кандидат, от вершин $v_i \in \{F\}$ вверх по дереву решения до достижения корневой вершины. Использование данного метода гарантирует поступление всех множеств подстановок, однозначно характеризующих возможные графы решения, если таковые имеются, в корневую вершину.

Основным недостатком изложенного метода параллельного логического вывода в процессорной структуре является отсутствие планирования загрузкой процессоров. Логический вывод относится к таким задачам, где степень параллелизма невозможно предсказать заранее, и, следовательно, сложно обеспечить равномерную загрузку процессоров во время проведения доказательства. Обычно перед системами параллельного логического вывода стоит проблема планирования загрузки процессоров. Решением может быть либо централизованное динамическое распределение процессов по процессорам, либо статическое и динамическое выявление параллелизма на основе анализа исходной базы знаний. Несмотря на то, что в процессорной структуре все процессорные модули, независимо от типа содержащейся в них вершин, принимают участие в поиске доказательства, полная нагрузка конкретных процессорных модулей будет зависеть только от частоты вхождения данной продукции в строящиеся графы решения. При этом наибольшую нагрузку будут нести вершины, осуществляющие операцию унификации, традиционно относящуюся к наиболее трудоемким операциям при логическом выводе.

Изложенный подход к поиску доказательства путем построения и анализа дерева решения ориентирован прежде всего на сильносвязные процессорные структуры [3]. То

обстоятельство, что коммутационная система активно участвует в поиске и извлечении решения, позволяет ускорить практически все этапы основного продукционного цикла.

ЛИТЕРАТУРА.

1. *Малпас Дж.* "Реляционный язык Пролог и его применение" / Москва, "Наука", 1990 г.
2. *Нильсон Н.* "Принципы искусственного интеллекта" / Москва, "Радио и связь", 1985 г.
3. *Каляев А.В., Кодачигов В.И.* "Гиперкубовые системы коммутации многопроцессорных систем" // НТЦ Интеграл, "Препринт" ИППММ, № 14-91, Львов, 1991 г.

УДК 658.512

В.И. Кодачигов, Л.К. Кодачигова

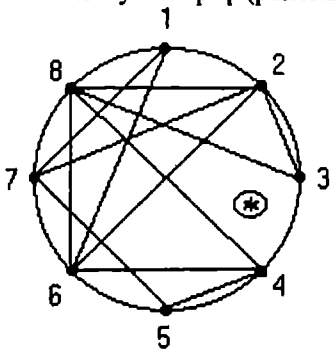
НЕКОТОРЫЕ СООБРАЖЕНИЯ ПО ПОВОДУ СВОЙСТВ И ВОЗМОЖНОСТЕЙ СТАНДАРТНЫХ ГРАФОВ

Пусть задан некоторый граф G со списком соединений:

1	6, 7
2	3, 6, 7, 8
3	2, 8
4	5, 6, 8
5	4, 7
6	1, 2, 4, 8
7	1, 2, 5
8	2, 3, 4, 6

Расположим вершины графа по окружности и на одинаковом , единичном расстоянии одна от другой. Перенумеруем их слева направо в естественном порядке

Получим граф (размещение) вида (*) :



Преобразуем теперь список соединений с учётом порядка смежности (считая смежность в порядке продвижения по направлению часовой стрелки).

Получим список G_*:

$G_*=$

1	6, 7
2	3, 6, 7, 8
3	2, 8
4	5, 6, 8
5	4, 7
6	1, 2, 4, 8
7	1, 2, 5
8	2, 3, 4, 6

Дальше поступим так. Попытаемся задать (связать) размещение с некоторой моделью , эталоном однозначно определяющей конкретное , данное размещение . В качестве таковой используемой представление графов в виде дерева.