УДК 681.3 + 519.68

ПРОБЛЕМЫ АППАРАТНО-ПРОГРАММНОЙ РЕАЛИЗАЦИИ ДЕДУКТИВНЫХ БАЗ ДАННЫХ В МНОГОПРОЦЕСОРНЫХ СИСТЕМАХ С РАСПРЕДЕЛЁННОЙ ПАМЯТЬЮ

Пушкаренко С.А., Дацун Н.Н.
Донецкий национальный технический университет

Рассматриваются проблемы распараллеливания процесса логического вывода в дедуктивных базах данных (ДБД) на многопроцессорных вычислительных системах (МВС) с распределенной памятью. В качестве средств моделирования использованы разработанный транслятор языка Дейталог, среда Visual C++ и библиотека MPI.

В параллельных системах с распределенной памятью используются интерфейсы, основанные на передаче сообщений, такие как MPI, PVM и др. Все эти интерфейсы способны осуществлять передачу линейных структур данных. В литературе, посвященной использованию интерфейсов передачи сообщений, обсуждаются в основном способы передачи именно таких структур данных [1, 2].

В логических языках программирования, используемых в ДБД в качестве языков запросов, все структуры данных нелинейны. В связи с этим возникает задача передачи нелинейных структур между процессорами МВС.

Наиболее удобной структурой данных для внутреннего представления логической программы является дерево. Дерево логического вывода представляет собой сильноветвящееся дерево. Каждый узел дерева может ссылаться на произвольное заранее неопределённое количество дочерних узлов, т.е. узлов нижнего уровня.

В дальнейшем под термином "дерево" будем понимать именно такое описанное выше сильноветвящееся дерево.

Узел дерева соответствует предикату. Порядок узлов в пределах уровня значения не имеет, т.к. используемый в исследовании язык Дейталог не чувствителен ни к порядку правил в программе, ни к порядку подцелей в теле правила [3].

Структура данных "дерево" может быть реализована различными способами [4]. Обозначим структуру данных "дерево" именем TREE. Ниже перечислены некоторые возможные варианта реализации дерева, представленного на рисунке 1:

1. одно из полей структуры TREE представляет собой указатель на начало списка указателей на дочерние узлы (см. рис. 2, а);

2.одно из полей структуры TREE является ссылкой на следующий узел того же уровня, т.е. на брата (см. рис. 2, б).

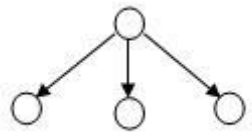
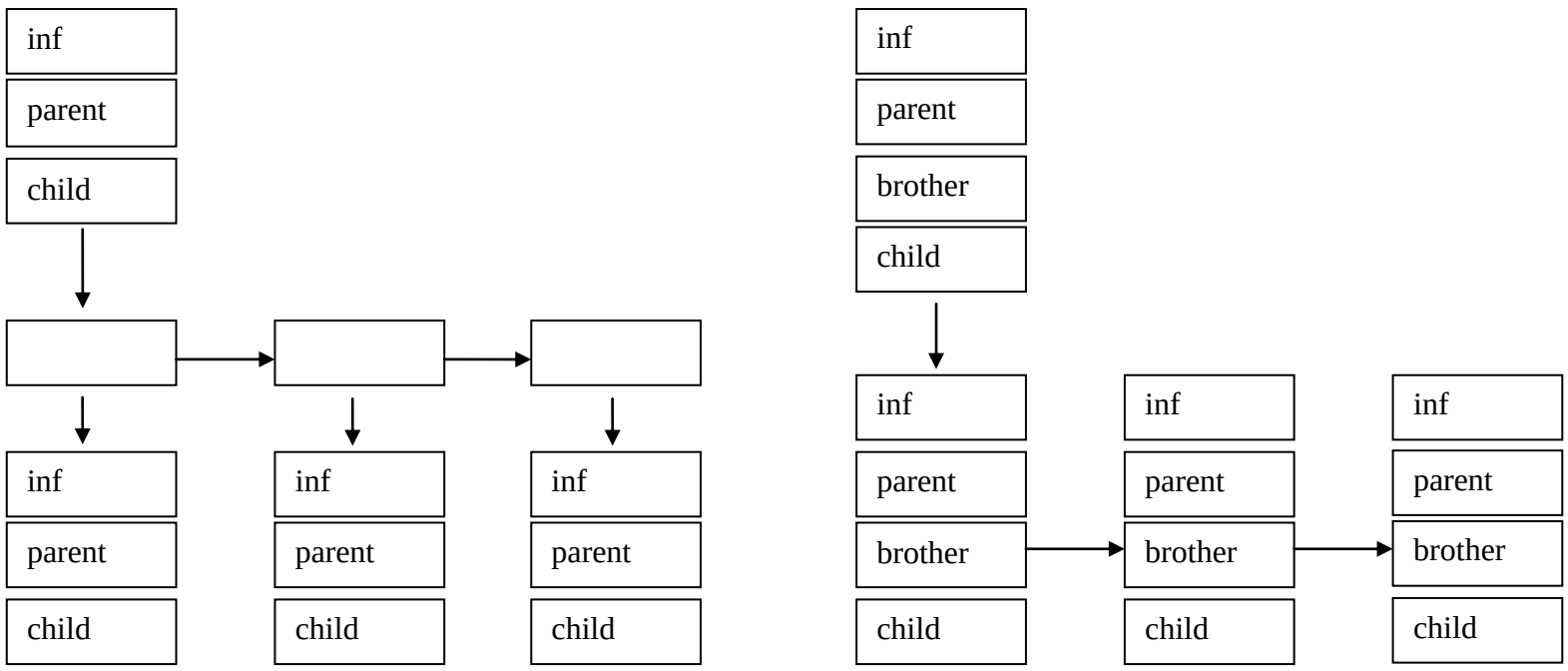


Рис.1. Пример дерева

Второй вариант представляется наиболее удобным как с точки зрения простоты реализации, так и с точки зрения работы с таким деревом. Так как в ДБД используется поиск сначала в ширину, а затем в глубину, то второй способ более предпочтителен:

нет необходимости каждый раз при просмотре очередного узла производить "откат" к родительскому узлу.



а) б)

Рис. 2. Схемы возможных способов представления дерева

Информационные поля содержат номер предиката и список его аргументов (номера переменных и констант). Так как число аргументов предиката величина заранее неизвестная, то аргументы удобнее хранить в структуре данных - "список".

При компиляции логической программы транслятор производит построение дерева вывода. На первом уровне дерева находятся узлы, соответствующие предикатам из головы правила, а нижние уровни содержат узлы, соответствующие предикатам из тела правила. Факты удобнее реализовать отдельным деревом или списком, либо же во время компиляции производить добавление записей, соответствующих фактам, в реляционную базу данных.

Ниже приведен алгоритм построения дерева:

Шаг 1. Компилятор анализирует логическую программу и выбирает очередной предикат из правила.

Шаг 2. Если выбранный предикат из головы правила, то просматривается первый уровень дерева. Если на первом уровне отсутствует узел, соответствующий данному предикату - создать узел и перейти на пункт 1. Иначе - перейти на пункт 3.

Если выбранный предикат из тела правила, то найти на первом уровне узел, соответствующий предикату из тела текущего правила и перейти на пункт 3.

Шаг 3. Для узла, соответствующего предикату из головы правила, создаются дочерние узлы, соответствующие предикатам из тела правила. При этом узлы, соответствующие предикатам, связанным между собой конъюнкцией, создаются на разных уровнях (являются потомком и родителем по отношению друг к другу), а узлы, соответствующие предикатам, связанным дизъюнкцией, находятся на одном уровне (являются "братьями").

Шаг 4. Если не конец программы - переход на пункт 1.

Когда пользователь даёт запрос, начинается процесс логического вывода, заключающийся в обходе дерева и выборе множества всех возможных кортежей, удовлетворяющих введенному пользователем запросу.

Процедура логического вывода, используемая в ДБД, является достаточно трудоёмкой задачей. Основной проблемой логического вывода является экспоненциальный рост пространства поиска [5].

Наиболее эффективным способом повышения производительности ДБД в настоящее время является распараллеливания процесса логического вывода [5].

Возможны следующие варианты реализации "распределённого" дерева логического вывода:

1.всем процессам передавать всё дерево;

2.каждому процессу передавать соответствующее поддерево.

Библиотека MPI позволяет передавать скалярные переменные и простые (линейные) структуры данных - массивы.

При работе с ДБД возникает необходимость передавать более сложные структуры данных - деревья. Для передачи дерева с помощью библиотеки MPI, необходимо преобразовать его в массив ("свернуть"), а "на обратной стороне" преобразовать этот массив обратно в исходное дерево ("развернуть"). Процесс преобразования сложных структур данных в линейные носит название "линеаризация".

Задача осложняется тем, что дерево является сильноветвящимся. Поэтому требуется передавать также структуру дерева.

Существует удобный и эффективный способ линеаризации дерева, заключающийся в следующем:

1. выбирается направление обхода дерева (сверху вниз, слева направо);

2. начинается обход дерева; при этом движение вперёд кодируется с помощью "1", движение назад с помощью "0".

Ниже описан разработанный алгоритм передачи дерева между процессорами.

Процедура "свёртки"/"развёртки" осуществляется в "интерактивном" режиме. Передающий процесс посылает сигнал "0" или "1", указывающий принимающему процессу направление движения (вперёд или назад).

"Развёртывание" дерева осуществляется с помощью конечного автомата с двумя состояниями ("0" и "1"). Начальное состояние - "1". Переход из одного состояния в другое происходит по сигналам от передающего процесса.

В состоянии "1" при получении сигнала "1" происходит ожидание данных от передающего процесса и после приёма данных осуществляется добавление дочернего узла к текущему узлу. При получении сигнала "0" происходит откат к родительскому узлу и переход автомата в состояние "0".

В состоянии "0" при получении сигнала "0" происходит откат к родительскому узлу и ожидание сигнала от передающего процесса. При получении сигнала "1" начинается ожидание данных от передающего процесса и, после приёма данных, происходит добавление ещё одного (нового) дочернего узла к текущему узлу и переход автомата в состояние "1".

В параллельной машине баз знаний, одной из задач которой является передача деревьев между процессорами, было бы целесообразно аппаратное реализовать функцию линеаризации дерева.

Литература

1. Шпаковский Г.И., Серикова П.В. Программирование для многопроцессорных систем в стандарте MPI. - Минск: БГУ, 2002. - 323 с.

2. Букатов А. А., Дацюк В. Н., Жегуло А. И.. Программирование многопроцессорных вычислительных систем. - Ростов-на-Дону. Издательство ООО «ЦВВР», 2003. - 208 с.

3. Чери С., Готлоб Г., Танка Л. Логическое программирование и базы данных: Пер. с англ. - М.: Мир, 1992. - 352 с.

3. Ахо Альфред В., Хопкрофт Джон, Ульман Джеффри Д. Структуры данных и алгоритмы.- М.: Издательский дом "Вильямс", 2000. - 384 с.

4. Вагин В.Н. и др. Достоверный и правдоподобный вывод в интеллектуальных системах /Под ред. В.Н. Вагина, Д.А. Поспелова. М.: Физматлитгиз, 2004. - 704 с.