УДК 004.43

ЭКСПЕРИМЕНТАЛЬНЫЕ ИССЛЕДОВАНИЯ СВОЙСТВ СИСТЕМЫ ПАРАЛЛЕЛЬНОГО ПРОГРАММИРОВАНИЯ

И.Л. Артемьева

Институт автоматики и процессов управления ДВО РАН
Россия, 690041, Владивосток, Радио ул., 5
E-mail: artemeva@iacp.dvo.ru

Н.Ю. Никифорова

Институт автоматики и процессов управления ДВО РАН
Россия, 690041, Владивосток, Радио ул., 5
E-mail: nikif@iacp.dvo.ru

М.Б. Тютюнник

Институт автоматики и процессов управления ДВО РАН
Россия, 690041, Владивосток, Радио ул., 5
E-mail: michaelhuman@gmail.com

Ключевые слова: параллельные вычисления, системы продукций, распараллеливание логического вывода
Key words: parallel computations, rule-based systems, parallel inference

В работе описаны экспериментальные исследования свойств системы параллельного программирования на различных наборах данных, показывающие эффективность примененных схем распараллеливания процесса выполнения правил и метода управления выбором схем.

EXPEREMENTAL RESEACH OF A PARALLEL SOFTWARE SYSTEM PROPERTIES / I.L. Artemieva (Institute for Automation and Control Processes, 5 Radio, Vladivostok, 690041, Russia, E-mail: artemeva@iacp.dvo.ru), N. YU. Nikiforova, (Institute for Automation and Control Processes, 5 Radio, Vladivostok, 690041, Russia, E-mail: nikif@iacp.dvo.ru), M.B. Tyutyunnik (Institute for Automation and Control Processes, 5 Radio, Vladivostok, 690041, Russia, E-mail: michaelhuman@gmail.com). The article describes the experimental study of parallel programming system properties. Experiments were conducted to evaluate the efficiency of the system operation with parallelization schemes and the method of control of scheme choice.

1. Введение

Продукционные языки и системы, основанные на правилах, являются важной технологией при создании информационных систем [15], называемых системами с базами правил (или с базами знаний, представленными в виде правил). При использовании таких

систем для решения реальных прикладных задач к настоящему времени разработаны базы знаний объемом в тысячи правил. В современных проектах по созданию Семантического Интернета предполагается использование правил как средств, расширяющих возможности языка для представления онтологий OWL (Web Ontology Language), позволяющих описывать бизнес-логику программных систем, создавать адаптеры между информационными системами и определять различные сложные запросы к информационным компонентам программных систем [15].

Если в системе, основанной на правилах, результат решения задачи зависит от порядка применения правил, то в механизме вывода неявно присутствует управление выбором правил при решении задачи либо в систему продукций вводятся явные средства управления процессом применения правил (т.е. средства описания алгоритма управления) [12]. В системах конфлюэнтных продукций [10] результат логического вывода не зависит от порядка применения правил, что не требует введения явных или неявных средств управления процессом решения задачи, позволяя рассматривать конфлюэнтные системы продукций не только как средство для представления знаний, но и как средство для задания метода решения задачи, т.е. как язык программирования метода.

Существует два основных способа организации вывода в системах продукций: прямой и обратный. При прямом выводе система использует информацию из антецедента правил, чтобы вывести информацию, описанную в консеквенте правил. При обратном выводе выдвигается цель - установить, можно ли вывести некоторый факт из системы продукций. К настящему времени разработаны методы оптимизации для прямого и обратного методов организации вывода, в том числе для конфлюэнтных систем продукций, причем наибольший эффект при оптимизации получен для систем последнего класса. Для систем, реализующих обратный вывод (в частности, систем, основанных на языке Пролог и его модификациях) [9], в литературе описаны методы организации параллельного вывода.

Однако существующие в настоящее время параллельные программные системы, основанные на правилах, базируются на неконфлюэнтных продукциях, поэтому для них разрабатываются сложные механизмы контроля процесса распараллеленного вывода. При этом при обратном выводе применение алгоритмов распараллеливания затрудняется наличием неявного управления порядком записи правил, порядком записи компонентов правил и порядком записи компонентов целевого утверждения, которое является входом системы, основанной на правилах. Распараллеливание прямого вывода учитывает лишь очень простые связи между правилами. Методы распараллеливания вывода, ориентированные на конфлюэнтные продукции, которые обладают естественным параллелизмом и не накладывают никаких дополнительных условий на управление процессом вывода, из литературы не известны.

Кроме того, в существующих программных системах, основанных на правилах, типы данных, используемые для моделирования объектов предметных областей, ограничены, в основном, предикатами, а сами языки являются языками с бедным семантическим базисом. В таких языках отсутствуют средства для компактного представления правил с похожей структурой. Наличие неявного управления выбором правил затрудняет сопровождение систем с большим количеством правил.

Выполненные ранее авторами исследования [3-8] привели к созданию системы, основанной на расширенной модели конфлюэнтных продукций, язык которой позволяет использование предикатных, функциональных и предметных имен, позволяет создание модульных программ и имеет средства компактного представления множеств правил. Построенные теоретические оценки времени выполнения программ при распараллеливании зависят от соотношения времени, требуемого правилу на обработку данных, к времени, требуемому на запуск параллельного процесса. Поэтому актуальными являются исследования этой зависимости как на модельных данных, так и на примерах приложений для реальных предметных областей и задач.

В работе описаны экспериментальные исследования свойств системы на различных наборах данных, которые показывают эффективность примененных схем распараллеливания процесса выполнения правил и метода управления выбором схем. В результате исследований было проведено 27 экспериментов, из которых 18 показывают работу приложений с разным строением информационного графа, 9 работу приложений для реальных предметных областей.

2. Экспериментальное исследование свойств системы параллельного программирования на модельных данных

Для экспериментов были выбраны логические программы, информационные графы которых являются элементарными, причем композиции таких программ дадут программу, имеющую информационный граф произвольного вида.

Эксперименты проводились на многопроцессорной кластерной ЭВМ с 70 доступными для вычислений процессорами, каждому процессору назначался один программный процесс. Характеристики узлов кластера следующие: процессор 2 x IBM PowerPC 970, частота 2.2GHz, память 4*1024Mb DDR-400, диск: 40Gb, управляющая сеть: Gigabit Ethernet, файловая сеть: Gigabit Ethernet, MPI сеть: Myrinet 2000 (4Gbps). На кластере установлена ОС Linux ppc64 (SuSE Enterprise Server 9), компилятор GNU C Compiler (3.3.3), MPI: MPICH-GM (1.2.6).

Клиент-серверная архитектура программы требует минимум 2 доступных процессора, один из которых занят диспетчерским процессом, остальные – выполнениями правил. При 2 доступных процессорах правила выполняются последовательно одно за другим.

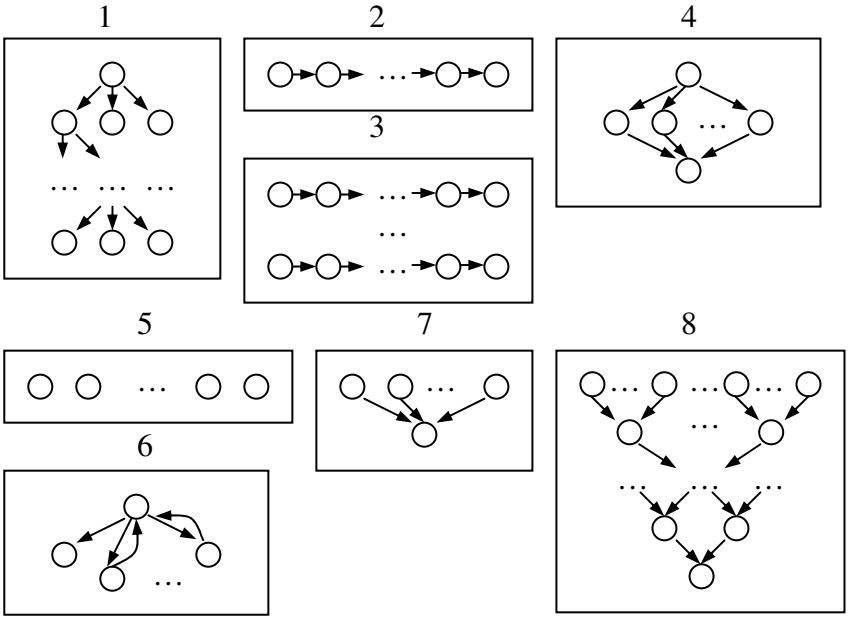


Рис. 1. Элементарные информационные графы

Было использовано девять элементарных информационных графов (рис. 1): (1) ИГ имеет одну начальную вершину и в каждую вершину (кроме начальной) входит одна дуга (рис. 1.1); (2) ИГ, в каждую вершину которого (за исключением начальной) входит единственная дуга, и из каждой вершины которого (за исключением конечной) входит единственная дуга, т.е. все вершины графа образуют цепочку, начинающуюся с начальной вершины и заканчивающуюся конечной (рис. 1.2); (3) ИГ содержит n независимых друг от друга цепочек вершин (рис. 1.3); (4) ИГ состоит из n вершин, причем единственная корневая

вершина соединена с (n-2) потомками, каждый из которых соединен дугой с единственной конечной вершиной (рис. 1.4); (5) ИГ содержит n независимых друг от друга одиночных вершин и не содержит дуг (рис. 1.5); (6) ИГ состоит из n вершин, (n-1) из которых образуют цикл (рис. 1.6); (7) единственная вершина ИГ соответствует правилу, содержащему префикс; (8) ИГ состоит из n вершин, (n-1) из которых являются начальными, причем все начальные вершины соединены дугой с единственной конечной вершиной (рис. 1.7); (9) ИГ состоит из n вершин, k из которых являются начальными (k<n), и одна – конечной; каждая i-ая вершина (кроме начальных) имеет m_i (m_i <n) предков и одного потомка (рис. 1.8).

Все эксперименты разбиты на группы, номер которой соответствует номеру типа элементарного информационного графа в приведенном выше списке. Для каждого типа графа в одной группе экспериментов создавалась одна или несколько программ. Программы отличаются количеством данных, передающихся между зависимыми между собой правилами. Таким образом, исследовалась зависимость времени работы программы от количества передаваемых данных между правилами и количества доступных процессоров.

Каждый эксперимент исполнялся несколько раз на одном и том же наборе входных данных, в описании результатов экспериментов приведено среднее время работы тестового приложения, которое равно сумме времен, деленных на количество тестовых прогонов. Следует отметить, что на скорость вычислений влияет время работы служебных процессов операционной системы, протокола MPI, задержки в коммуникациях. Особенно это проявляется в том случае, когда время выполнения правил маленькое (десятые доли секунды).

Описание эксперимента группы 1

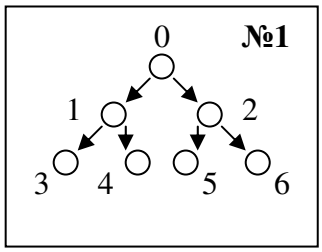
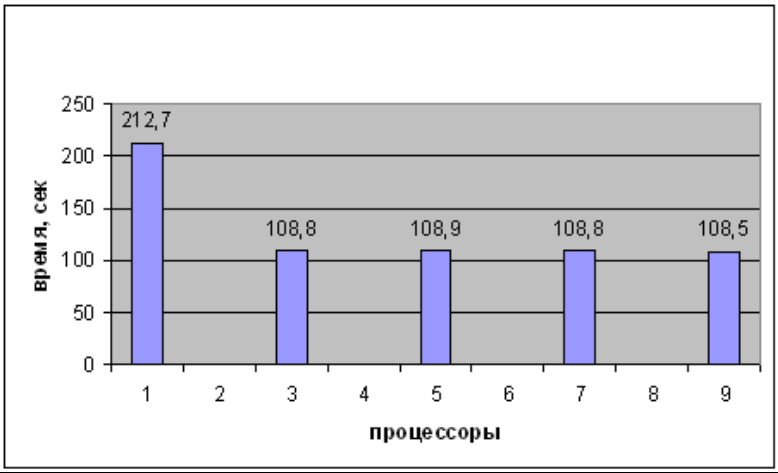


Рис. 2. ИГ №1 программы с 7 правилами

Для данного эксперимента была выбрана логическая программа, состоящая из 7 правил. Информационный граф программы показан на рисунке 2 и представляет собой дерево, в котором каждая вершина имеет одного предка (кроме корневой вершины «0») и двух потомков (кроме конечных вершин «3», «4», «5», «6»). Из правил видно, что все три объекта-отношения, входящие в условие одного правила, входят в следствие правила, от которого зависит данное правило.



Время работы программы показано на диаграмме 1. Максимальное время выполнения программы, когда каждое правило выполнялось последовательно, составило 212,7 сек. при 1 рабочем процессе, минимальное – 108,5 сек. В параллельном режиме эффективность вычислений возросла в среднем в 1,96 раза при 3 рабочих процессах.

Согласно методу управления выбором схем в эксперименте в процессе логического вывода применяются две схемы выполнения правил: схема распараллеливания правил с использованием информационного графа и схема с передачей кортежей при неполном выполнении правила.

Описание эксперимента группы 2

Эксперимент 1. Для данного эксперимента была выбрана логическая программа, состоящая из 7 правил. Информационный граф программы показан на рисунке 3, из которого видно, что все правила образуют цепочку, в которой каждая вершина (кроме корневой вершины «0») имеет одного предка и одного потомка (кроме конечной вершины). Из правил видно, что все три объекта-отношения, входящие в условие одного правила, входят в следствие другого правила, от которого зависит данное правило.

Время работы программы показано на диаграмме 2. Максимальное время выполнения программы, когда каждое правило выполнялось последовательно, составило 140,6 сек., минимальное – 26,6 сек. В параллельном режиме эффективность вычислений возросла в среднем в 5,28 раза при 7 рабочих процессах.

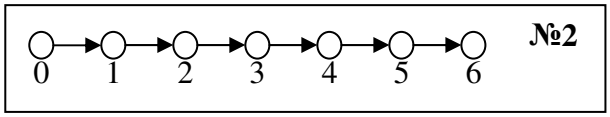


Рис. 3. ИГ №2 программы с 7 правилами

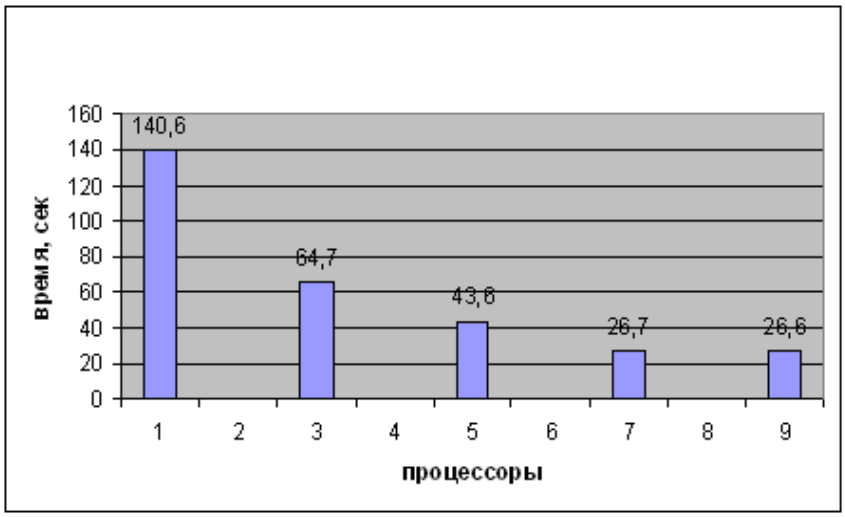


Диаграмма 5.2.1. Время выполнения программы, эксперимент 1

Согласно методу управления выбором схем в эксперименте в процессе логического вывода применяется только одна схема с передачей кортежей при неполном выполнении правила.

Описание эксперимента группы 3

Эксперимент 1. Для данного эксперимента была выбрана логическая программа, состоящая из 12 правил. Информационный граф программы показан на рисунке 4. Он содержит 4 независимых цепочки вершин. Из правил видно, что все три объекта, входящие в условие одного правила, входят в следствие правила, от которого зависит данное правило.

Время работы программы показано на диаграмме 3. Максимальное время выполнения программы, когда каждое правило выполнялось последовательно, составило 365,2 сек.,

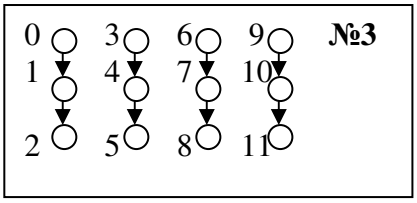


Рис. 4. ИГ №3 программы с 12 правилами

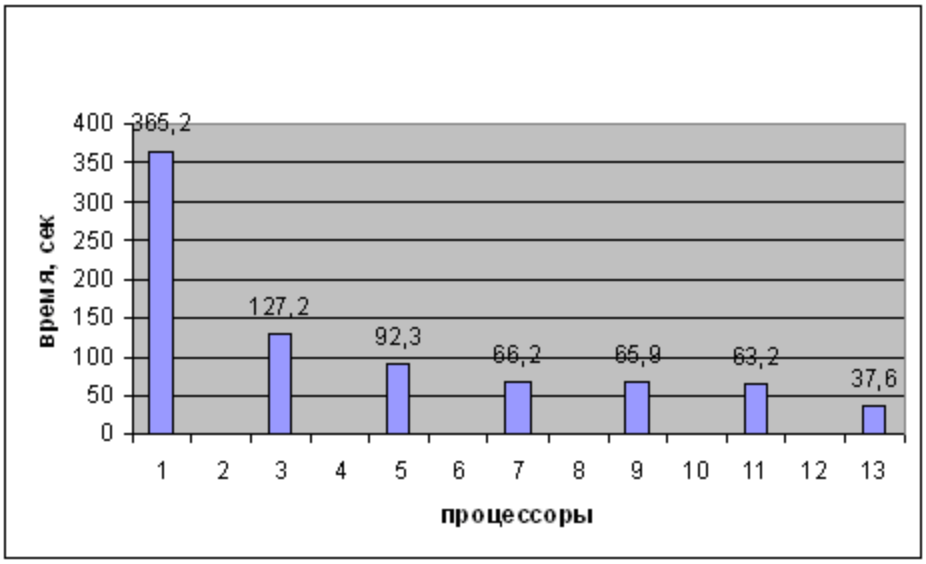


Диаграмма 3. Время выполнения программы, эксперимент 1

минимальное – 37,6 сек. В параллельном режиме эффективность вычислений возросла в среднем в 9,71 раза.

Согласно методу управления выбором схем в эксперименте в процессе логического вывода применяются две схемы выполнения правил: схема распараллеливания правил с использованием информационного графа и схема с передачей кортежей при неполном выполнении правила.

Описание эксперимента группы 4

Эксперимент 1. Для данного эксперимента была выбрана логическая программа,

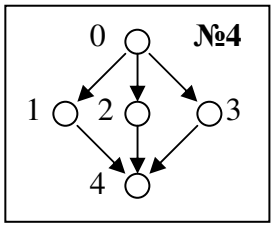


Рис. 5. ИГ №4 программы с 5 правилами

состоящая из 5 правил. Информационный граф программы показан на рисунке 5. В графе начальная вершина имеет 3 потомков, а конечная – трех предков. Из правил видно, что все три объекта, входящие в условие одного правила, входят в следствие правила, от которого зависит данное правило.

Время работы программы показано на диаграмме 4. Максимальное время выполнения программы, когда каждое правило выполнялось последовательно, составило 154,1 сек., минимальное – 134,5 сек. В параллельном режиме эффективность вычислений возросла в среднем в 1,14 раза.

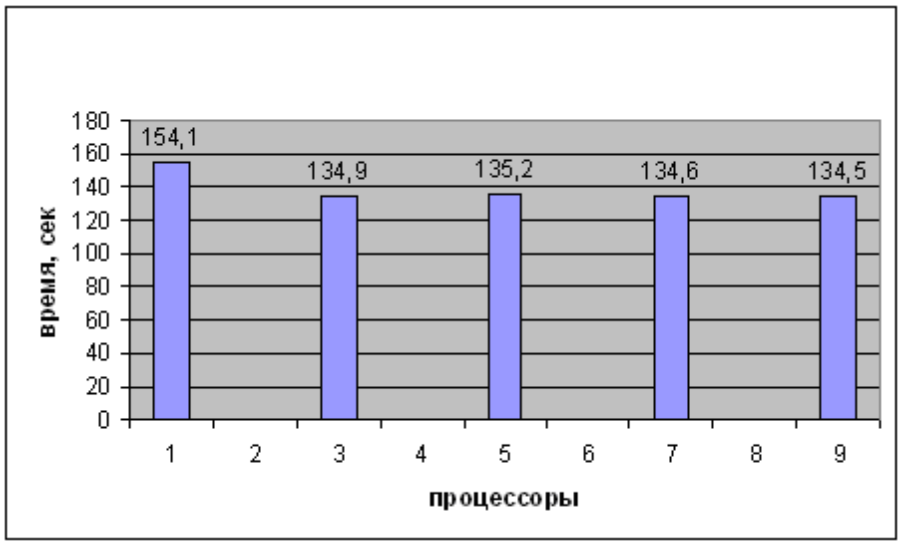


Диаграмма 4. Время выполнения программы, эксперимент 1

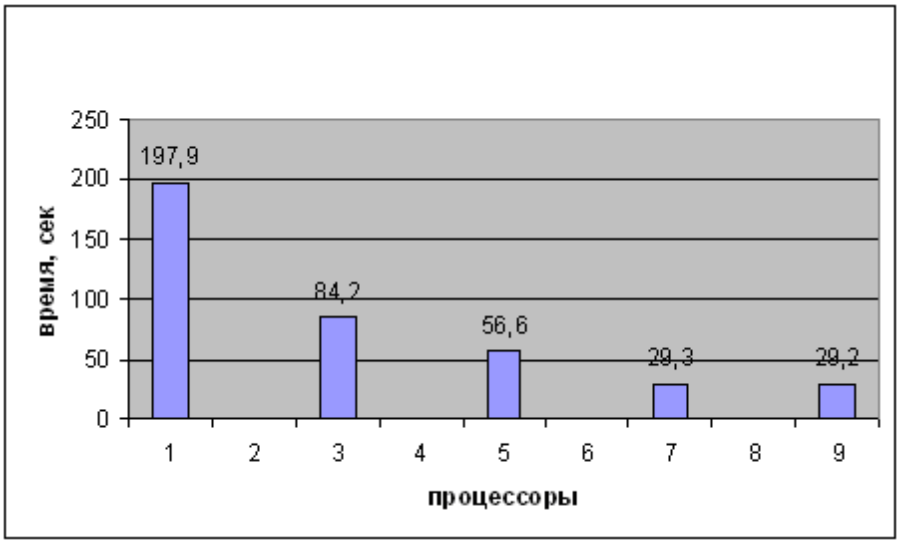


Диаграмма 5. Время выполнения программы, эксперимент 1

Согласно методу управления выбором схем в эксперименте в процессе логического вывода применяются две схемы выполнения правил: схема распараллеливания правил с использованием информационного графа и схема с передачей кортежей при неполном выполнении правила.

Описание эксперимента группы 5

Эксперимент 1. Для данного эксперимента была выбрана логическая программа, состоящая из 7 правил. Информационный граф программы показан на рисунке 6. Он содержит 7 независимых друг от друга вершин и не имеет дуг.

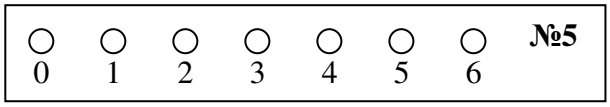


Рис. 6. ИГ №5 программы с 7 правилами

Время работы программы показано на диаграмме 5. Максимальное время выполнения программы, когда каждое правило выполнялось последовательно, составило 197,9 сек., минимальное – 29,2 сек. В параллельном режиме эффективность вычислений возросла в среднем в 6,77 раза.

Согласно методу управления выбором схем в эксперименте в процессе логического вывода применяется схема распараллеливания правил с использованием информационного графа.

Описание эксперимента группы 6

Эксперимент 1. Для данного эксперимента была выбрана логическая программа, состоящая из 3 правил. Информационный граф программы показан на рисунке 7. Они содержит цикл из двух вершин.

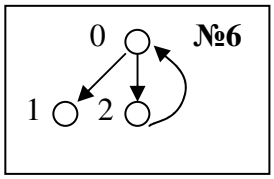


Рис. 7. ИГ №6 программы с 3 правилами

Время работы программы показано на диаграмме 6. Максимальное время выполнения программы, когда каждое правило выполнялось последовательно, составило 53,3 сек., минимальное – 41,4 сек. В параллельном режиме эффективность вычислений возросла в среднем в 1,3 раза.

Согласно методу управления выбором схем в процессе логического вывода применяются две схемы выполнения правил: схема распараллеливания правил с

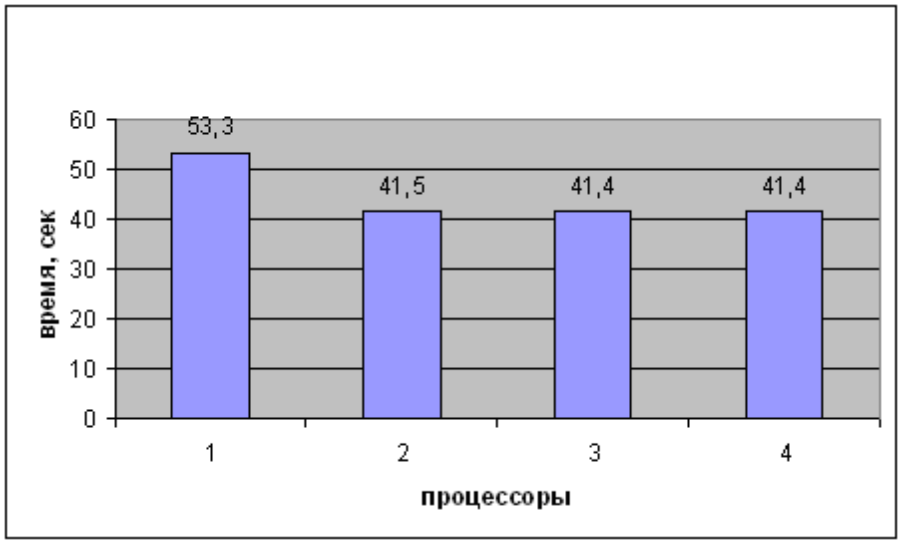


Диаграмма 6. Время выполнения программы, эксперимент 1

использованием информационного графа и схема с передачей кортежей при неполном выполнении правила.

Описание эксперимента группы 7

Эксперимент 1. Для данного эксперимента была выбрана логическая программа, состоящая из одного правила, содержащего префикс. Информационный граф программы состоит из одной вершины. Префикс содержит два индекса переменных.

Время работы программы показано на диаграмме 7.

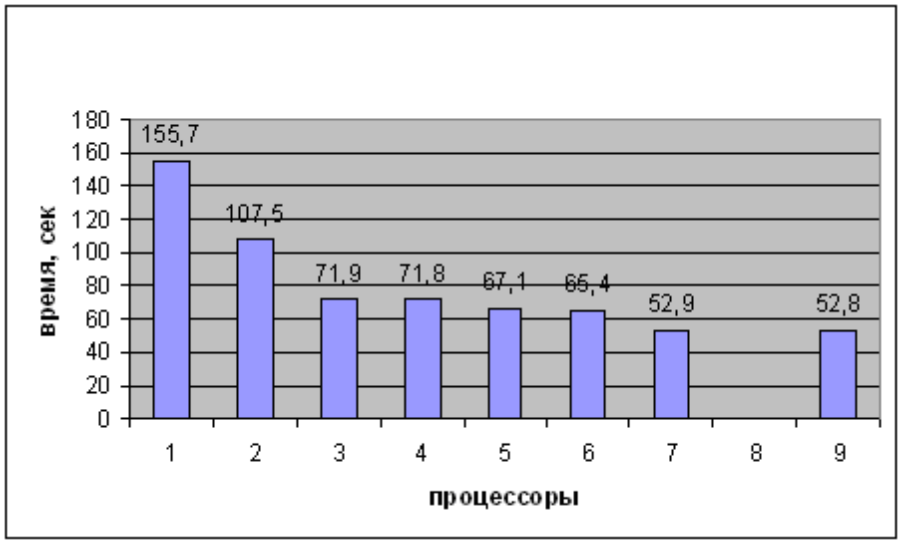


Диаграмма 7. Время выполнения программы, эксперимент 1

Максимальное время выполнения программы, когда каждое правило выполнялось последовательно, составило 155,7 сек., минимальное – 52,8 сек. В параллельном режиме эффективность вычислений возросла в среднем в 2,94 раза.

Согласно методу управления выбором схем в эксперименте в процессе логического вывода применяется только одна схема распараллеливания вычислений для правил с префиксом.

В эксперименте время работы программы в параллельном режиме оказалось меньше, чем в последовательном. Таким образом, применение схемы распараллеливания позволило снизить время работы приложения.

Описание эксперимента группы 8

Эксперимент 1. Для данного эксперимента была выбрана логическая программа, состоящая из 4 правил. Информационный граф программы показан на рисунке 8. В графе 3 начальных вершины имеют одного общего потомка. Из правил видно, что все три объекта, входящие в условие одного правила, входят в следствие правила, от которого зависит данное правило.

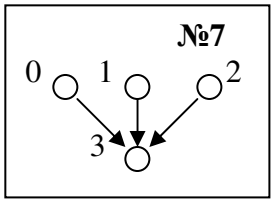


Рис. 8. ИГ №7 программы с 4 правилами

Время работы программы показано на диаграмме 8. Максимальное время выполнения программы, когда каждое правило выполнялось последовательно, составило 96,4 сек.,

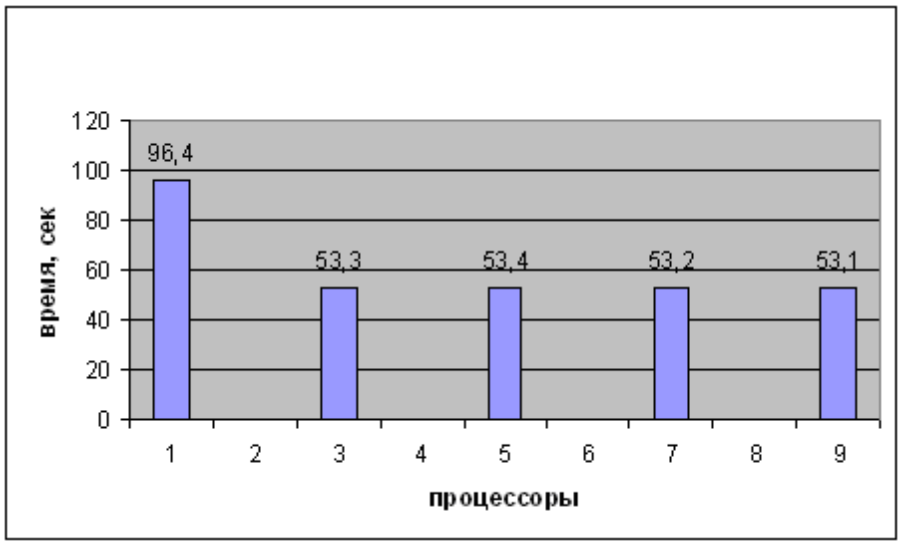


Диаграмма 8. Время выполнения программы, эксперимент 1

минимальное – 53,1 сек. В параллельном режиме эффективность вычислений возросла в среднем в 1,81 раза.

Согласно методу управления выбором схем в процессе логического вывода применяется одна схема выполнения правил: схема распараллеливания правил с использованием информационного графа. В эксперименте время работы программы в параллельном режиме оказалось меньше, чем в последовательном. Таким образом, применение схемы распараллеливания позволило снизить время работы приложения.

Описание эксперимента группы 9

Эксперимент 1. Для данного эксперимента была выбрана логическая программа, состоящая из 7 правил. Информационный граф программы показан на рисунке 9. В графе каждая вершина (кроме начальных) имеет двух предков. Из структуры правил видно, что все три объекта, входящие в условие одного правила, входят в следствие правила, от которого зависит данное правило.

Время работы программы показано на диаграмме 9. Максимальное время выполнения программы, когда каждое правило выполнялось последовательно, составило 129,9 сек., минимальное – 64,7 сек. В параллельном режиме эффективность вычислений возросла в среднем в 2 раза.

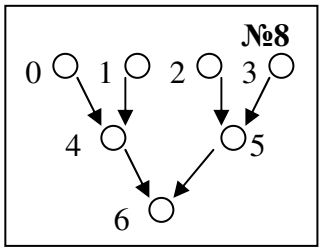


Рис. 9. ИГ №8 программы с 7 правилами

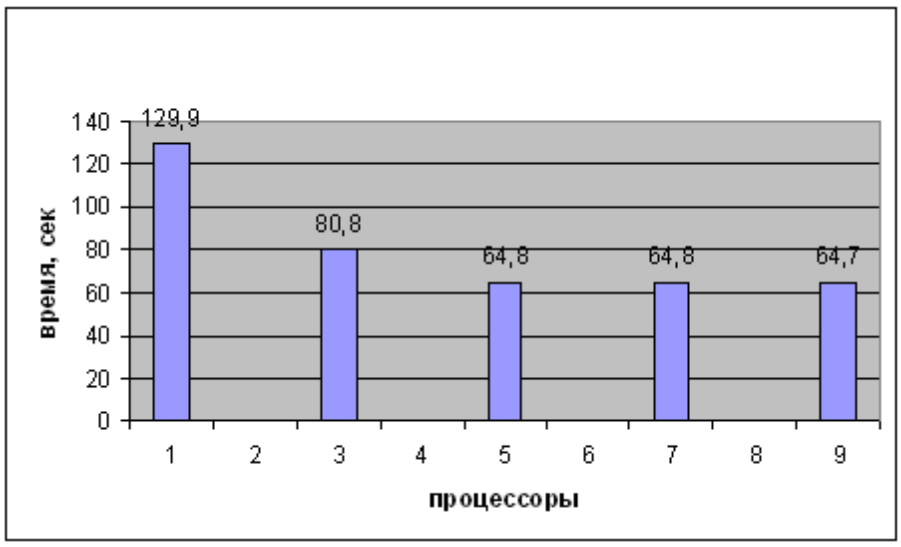


Диаграмма 9. Время выполнения программы, эксперимент 1

Согласно методу управления выбором схем в эксперименте в процессе логического вывода применяется одна схема выполнения правил: схема распараллеливания правил с использованием информационного графа. В эксперименте время работы программы в параллельном режиме оказалось меньше, чем в последовательном. Таким образом, применение схемы распараллеливания позволило снизить время работы приложения.

3. Экспериментальное исследование системы параллельного программирования на примерах реальных задач

Для того чтобы исследовать систему параллельного программирования на реальных примерах, были выбраны задачи из разных предметных областей: медицины, вычислительной математики, химии. В следующих пунктах описаны разработанные программные системы и результаты их исследований.

Системы решения упрощенной задачи медицинской диагностики. Случай: для заболеваний задано одно клиническое проявление

На основе упрощенной онтологии медицины [1,2] была разработана программная система, решающая задачу медицинской диагностики и состоящая из редактора базы знаний, системы ввода исходных данных и решателя задач диагностики методом опровержения гипотез. Упрощенная модель онтологии описывает случай, когда пациент болен только одним заболеванием, причем заболевание рассматривается как процесс, протекающий во времени. Для каждого заболевания в базе знаний описан только один вариант клинического проявления. Признаки, с помощью которых описывается состояние пациента, также рассматриваются как процессы, протекающие во времени. Каждый процесс-признак имеет периоды, характеризующиеся множествами значений признаков и имеющими определенную длительность у пациента, причем у разных пациентов длительность периодов проявления значений признаков может быть разной, но возможная длительность каждого периода определяется заболеванием, которым болен пациент (что задается в знаниях). При решении задачи делаются попытки опровергнуть гипотезы обо всех возможных заболеваниях на основе исходных данных. Те гипотезы, которые не удается опровергнуть, образуют результат процесса опровержения – множество возможных диагнозов для пациента.

Редактор базы знаний и система ввода исходных данных о значениях признаков, наблюдавшихся у пациента в разные моменты наблюдения являются внешними программными средствами, разработанными студентами ДВГУ при выполнении учебных практических работ по дисциплине "Системы искусственного интеллекта". Каждое программное средство заносит введенные с его помощью данные в базу данных, которая затем используется решателем прикладной задачи. С помощью данных средств были подготовлены наборы входных данных для двух решателей задачи медицинской диагностики, созданных с использованием системы параллельного программирования. В ходе экспериментов изучались временные характеристики обеих решателей.

Эксперимент 1

В рамках данного эксперимента был создан первый решатель задачи медицинской диагностики, состоящий из 4 логических модулей, которые в сумме описывают 20 правил. Первые три модуля содержат правила, подготавливающие промежуточные данные на основе базы знаний о заболеваниях и входных данных о пациенте, четвертый модуль реализует метод опровержения гипотез. Первый модуль состоит из 4 правил, второй – из 7, третий – из 1, четвертый – из 8, причем 18 правил содержат префиксы, которые задают области значений от одной до 4 переменных. В правилах также используются кванторные циклы.

Наличие префиксов у правил позволяет в процессе логического вывода применить схему распараллеливания выполнения правил с префиксом. Также используется метод распараллеливания на основе свойств информационного графа.

База знаний, подготовленная с использованием редактора знаний, содержит знания о 4 заболеваниях, в ней описаны 6 признаков со своими областями значений, для заболеваний указаны признаки, входящие в их клинические картины, определены периоды динамики признаков при заболеваниях, границы их длительности и значения признаков в каждый период.. С использованием системы ввода исходных данных заданы значения разных признаков в 9 различных моментах наблюдения. Время работы первого решателя для разного числа процессоров в испытании для данного набора данных приведено в таблице 1.

При последовательном выполнении модулей (число рабочих процессоров равно 1) время работы составило 12,3 секунды. При

Табл. 1. Время работы первого решателя задачи в ходе эксперимента 1

Число процессоров	**1**	**2**	**4**	**6**	**9**	**29**
Модуль 1	1,7 сек	1,7 сек	1,7 сек	2,1 сек	2,2 сек	2,2 сек
Модуль 2	6,3 сек	8,9 сек	10,3 сек	10,5 сек	10,8 сек	11,8 сек
Модуль 3	0,8 сек	0,8 сек	0,1 сек	0,9 сек	0,9 сек	0,8 сек
Модуль 4	3,5 сек	4,4 сек	5,3 сек	5,6 сек	5,6 сек	5,9 сек
Суммарное время	12,3 сек	15,8 сек	17,4 сек	19,1 сек	19,5 сек	20,7 сек

параллельном режиме при увеличении числа процессоров время работы модулей увеличилось, при 29 процессорах время работы составило 20,7 секунды. Это связано с тем, что во втором и четвертом модулях правила содержат префиксы, для каждого значения переменной из которых создаются копии правил, обрабатываемые параллельно в соответствии со схемой распараллеливания для правил с префиксом. При этом время работы каждого правила существенно меньше времени, затраченного на подготовку и пересылку данных между процессами, обрабатывающими правила, то есть затраты на системные вызовы и избыточный параллелизм превысили затраты на выполнение правил.

Эксперимент 2

Вторая система использует редактор знаний и систему ввода данных, описанные выше, и оптимизированный решатель задачи медицинской диагностики, который содержит 3 логических модуля и 10 правил: первые два модуля содержат правила, подготавливающие промежуточные данные на основе базы знаний о заболеваниях, а третий реализует метод опровержения гипотез и является модификацией модуля 4 предыдущего примера. Первые два модуля содержат по 1 правилу, а третий – 8 правил. Все правила третьего модуля имеют префиксы, которые задают области значений только для одной переменной.

Экспериментальные исследования проводились на двух наборах данных. Первый набор полностью совпадает с набором данных, использованных в эксперименте 1. Время работы второго решателя задачи медицинской диагностики для разного числа процессоров приведено в таблице 2.

Табл. 2. Время работы второго решателя задачи в ходе эксперимента 2 на первом наборе данных

Число Процессоров	**1**	**2**	**6**	**9**	**29**
Модуль 1	0,4 сек	0,4 сек	0,9 сек	0,9 сек	0,9 сек
Модуль 2	0,8 сек	0,8 сек	0,8 сек	0,8 сек	0,8 сек
Модуль 3	0,5 сек	0,8 сек	1,6 сек	1,6 сек	1,7 сек
Суммарное время	1,7 сек	2 сек	3,3 сек	3,3 сек	3,4 сек

При последовательном выполнении модулей (число рабочих процессоров равно 1) время работы составило 1,7 секунды. При параллельном режиме при увеличении числа процессоров время работы модулей увеличилось незначительно, при 29 процессорах время работы составило 3,4 секунды. Увеличение времени связано с временными затратами на системную обработку большого количества процессов относительно малого времени выполнения отдельных правил.

Сравним время работы первого и второго решателей задачи медицинской диагностики на первом наборе данных. Время работы второго решателя по сравнению с первым для последовательного режима уменьшилось в 7,2 раза, для параллельного режима при 29 процессорах – в 6 раз. Время работы 3 модуля второго решателя по сравнению со временем работы 4 модуля первого решателя для последовательного режима уменьшилось в 7 раз[1], для параллельного режима при 29 процессорах – в 3,4 раза. Уменьшение времени работы объясняется как уменьшением общего числа правил, так и сокращением количества переменных, описанных в префиксах правил.

Для данного эксперимента был подготовлен второй, расширенный набор данных. Второй набор входных данных содержит знания о 11 заболеваниях, 16 признаках, 66 записей о клинической картине, 170 значениях признаков, а также значения признаков для 169 моментов наблюдения. Время работы второго решателя для разного числа процессоров в испытании для второго набора данных приведено в таблице 3.

Табл. 3. Время работы второго решателя задачи медицинской диагностики на втором наборе данных

Число процессоров	**1**	**2**	**4**	**6**	**9**	**29**
модуль 1	0,3 сек	0,4 сек	0,6 сек	0,6 сек	0,6 сек	0,8 сек
модуль 2	0,9 сек	0,9 сек	0,9 сек	0,9 сек	0,9 сек	0,9 сек
модуль 3	34,7 сек	22,3 сек	15,7 сек	12,5 сек	11,4 сек	11,1 сек

[1] Заметим, что оба модуля используют при выполнении один и тот же набор входных данных

Суммарное время	35,9 сек	23,6 сек	17,2 сек	14 сек	12,9 сек	12,8 сек

При последовательном выполнении модулей (число рабочих процессоров равно 1) время работы составило 35,9 секунды. При параллельном режиме при увеличении числа процессоров время работы модулей уменьшилось, при 29 процессорах время работы составило 12,8 секунды. По сравнению с последовательным режимом уменьшение в 2,5 раза на 6 рабочих процессорах, в 2,8 раза при 29 процессорах. Уменьшение времени связано с увеличением времени работы отдельных правил (увеличился объем обрабатываемых данных).

Были также проведены испытания на втором наборе данных для 4 модуля первого решателя задачи медицинской диагностики (см. таблицу 4).

Табл. 4. Время работы четвертого модуля первого решателя задачи медицинской диагностики на втором наборе данных

Число процессоров	**1**	**2**	**4**	**6**	**9**	**29**
модуль 1 (M2)	0,3 сек	0,4 сек	0,6 сек	0,6 сек	0,6 сек	0,8 сек
модуль 2 (M2)	0,9 сек	0,9 сек	0,9 сек	0,9 сек	0,9 сек	0,9 сек
модуль 4 (M1)	84,5 сек	118,6 сек	130,4 сек	138,7 сек	140,8 сек	147,2 сек
Суммарное время	85,7 сек	119,9 сек	131,9 сек	140,2 сек	142,3 сек	148,9 сек

При последовательном выполнении модулей (число рабочих процессоров равно 1) время работы составило 85,7 секунды. При параллельном режиме при увеличении числа процессоров время работы модулей увеличилось, при 29 процессорах время работы составило 148,9 секунды. Увеличение времени работы объясняется тем, что время работы каждого параллельного процесса существенно меньше времени, затраченного на подготовку и пересылку данных между процессами, то есть затраты на системные вызовы и избыточный параллелизм превысили затраты на выполнение правил.

Сравним время работы 3 модуля второго решателя и 4 модуля первого решателя, которые реализуют метод опровержения гипотез. Время работы 3 модуля второго решателя по сравнению со временем 4 модуля первого решателя для последовательного режима уменьшилось в 2,4 раза, для параллельного режима при 29 процессорах – в 13,2 раза. Уменьшение времени работы объясняется сокращением количества переменных, описанных в префиксах правил, что позволило увеличить время выполнения каждого параллельного процесса и уменьшить соотношение между временем работы процесса и временем, требуемым на системные вызовы.

Системы решения упрощенной задачи медицинской диагностики. Случай: для заболеваний задано несколько клинических проявлений

Эксперимент 1

В рамках данного эксперимента был создан третий решатель задачи медицинской диагностики методом опровержения гипотез, построенный на основе упрощенной онтологии медицинской диагностики, которая описывает случай, когда пациент болен только одним заболеванием, причем заболевание рассматривается как процесс, протекающий во времени. Для каждого заболевания в базе знаний описано несколько вариантов клинического проявления, а также необходимое условие заболевания. Признаки также рассматриваются как процессы, протекающие во времени. Признаки могут быть как простыми, так и состоящими из характеристик. Каждая характеристика-признак и каждый признак-процесс

имеет периоды, характеризующиеся множествами значений и имеющими определенную длительность у пациента (свои для каждого варианта клинического проявления). При решении задачи делаются попытки опровергнуть гипотезы о варианте клинического проявления. Гипотеза о заболевании опровергается, если опровергнуты гипотезы обо всех вариантах его клинического проявления. Те гипотезы о заболевании, которые не удается опровергнуть, образуют результат процесса опровержения – множество возможных диагнозов для пациента.

Для задания входных данных, описывающих значения признаков, характеристик, наблюдавшихся у пациента в разные моменты времени, было создано внешнее программное средство «Ввод исходных данных». Программное средство заносит введенные с его помощью данные в базу данных, которая затем используется решателем прикладной задачи. С помощью данного средства были подготовлены наборы входных данных для трех решателей задачи медицинской диагностики, созданных с использованием системы параллельного программирования.

Метод решения задачи описывается одним модулем, содержащим 19 правил опровержения. Все правила имеют префикс, описывающий область значений (название заболевание) для одной переменной, а также используют кванторные циклы. Наличие префиксов у правил позволяет в процессе логического вывода применить не только схему распараллеливания с использованием информационного графа, но и схему распараллеливания для правил с префиксом.

Для данного решателя были подготовлены два набора данных. При подготовке первого набора были использованы формально представленные знания о 6 хирургических заболеваниях органов брюшной полости, подготовленные экспертом в области медицины М.Ю. Черняховской для экспертной системы Консультант-2 [13,14]. Этот набор содержит знания о 6 заболеваниях острого живота, 60 признаках, 121 характеристике признаков, 215 записей о клинической картине заболеваний, 282 записей о значениях характеристик признаков, 382 записей о числе периодов динамики характеристик при заболеваниях, 222 записей о числе вариантов значений характеристик, 209 записей о нижней границе и 209 записей о верхней границе для вариантов клинического проявления характеристик при заболеваниях, а также данные о значениях характеристик 2 признаков при 2 моментах их наблюдения у пациента. Результаты решения задачи совпали с результатами, полученными при работе экспертной системы Консультант-2. Время работы решателя для разного числа процессоров в испытании для первого набора данных приведено в таблице 5.

Табл. 5. Время работы третьего решателя задачи медицинской диагностики в ходе эксперимента 1 на первом наборе данных

Число процессоров	**1**	**2**	**4**	**6**	**9**	**29**
модуль 1	4,5 сек	3,2 сек	2,6 сек	2,6 сек	2,6 сек	2,7 сек

При последовательном выполнении модулей (число рабочих процессоров равно 1) время работы составило 4,5 секунды. При параллельном режиме при увеличении числа процессоров время работы модулей уменьшилось, минимальное время равно 2,7 секунды. По сравнению с последовательным режимом уменьшение в 1,7 раза при числе процессоров больше 3. Уменьшение времени связано с работой схемы распараллеливания, использующих информационный граф, и схемы распараллеливания правил с префиксом, а также с небольшим числом значений переменных, описанных в префиксах (каждый префикс описывает переменную, область значений которой состоит из 6 заболеваний).

Также был подготовлен второй, расширенный набор модельных данных. Второй набор входных данных содержит дополнительные, по сравнению с первым набором, знания о 48 значениях характеристик 14 признаков при 2 моментах наблюдения, 500 записей о нижней границе и 500 записей о верхней границе для вариантов наблюдений значений

характеристик. Время работы решателя для разного числа процессоров в испытании для второго набора данных приведено в таблице 6.

Табл. 6. Время работы третьего решателя задачи медицинской диагностики на втором наборе данных

Число процессоров	**1**	**2**	**4**	**6**	**9**	**29**
Модуль 1	27,2 сек	16,2 сек	10,1 сек	6,9 сек	6,9 сек	6,9 сек

При последовательном выполнении модулей (число рабочих процессоров равно 1) время работы составило 27,2 секунды. При параллельном режиме при увеличении числа процессоров время работы модулей уменьшилось, при 29 процессорах время работы составило 6,9 секунды. По сравнению с последовательным режимом уменьшение - в 3,9 раза при числе процессоров больше 5.

Сравним результаты эксперимента, проведенного на двух наборах данных. Время работы решателя на втором наборе данных по сравнению со временем работы на первом наборе для последовательного режима увеличилось в 6 раз, для параллельного режима при 6 и более процессорах – в 2,6 раза. Можно отметить, что на втором наборе данных при увеличении числа процессоров эффективность вычислений возрастает. Это связано с тем, что время работы решателя на первом наборе данных сравнительно небольшое и на него накладываются системные задержки.

Эксперимент 2

Четвертый решатель задачи медицинской диагностики был получен из третьего следующим образом. Переменная, принимающая значение заданного признака, была вынесена из тела каждого правила в его префикс. Таким образом, число переменных в каждом префиксе было увеличено с одного до двух, что предполагает дополнительное распараллеливание правил с помощью схемы распараллеливания правил с префиксом.

Для данного эксперимента был использован второй набор данных из эксперимента 1. Время работы решателя для разного числа процессоров в приведено в таблице 7.

При последовательном выполнении модулей (число рабочих процессоров равно 1) время работы составило 55,8 секунды. При параллельном режиме при увеличении числа процессоров время работы модулей уменьшилось при числе процессоров до 6 (49,8 секунды при 2

Табл. 7. Время работы решателя задачи медицинской диагностики в ходе эксперимента 1 на втором наборе данных

Число процессоров	**1**	**2**	**4**	**6**	**9**	**29**
модуль 1	55,8 сек	49,8 сек	53,6 сек	55,2 сек	57,7 сек	61,9 сек

процессорах), при 29 процессорах время работы увеличилось и составило 61,9 секунды. Это связано с тем, что правила содержат префиксы, для каждого значения переменной которых создаются копии правил и обрабатываются параллельно в соответствии со схемой распараллеливания правил с префиксом, при этом количество параллельных вызовов процессоров превышает число доступных процессоров. Время работы каждого правила меньше времени, затраченного на подготовку и пересылку данных между процессами, обрабатывающими правила, то есть затраты на системные вызовы и избыточный параллелизм превысили затраты на выполнение правил.

Сравним результаты экспериментов 1 и 2 на втором наборе данных. Время работы третьего решателя по сравнению со временем работы четвертого решателя для последовательного режима увеличилось в 2 раза, для параллельного режима при 29

процессорах – в 8,9 раза. Увеличение времени вычислений связано с увеличением числа переменных в префиксах.

Эксперимент 3

В данном эксперименте для демонстрации возможностей входного языка системы параллельного программирования были разработано две программные системы: первая представляет собой компилятор конкретизаций правил, а вторая – решатель, использующий скомпилированные конкретизации и описанные заранее схемы правил при решении задачи диагностики методом опровержения гипотез.

Компилятор конкретизаций правил состоит из одного модуля, где единственное правило генерирует список конкретизаций правил для всех заболеваний (упрощенная компиляция правил по базе знаний). Решатель задачи медицинской диагностики для раздела медицины, задаваемого списком заболеваний, состоит из одного модуля, содержащего 19 схем правил и подгружающего список сгенерированных конкретизаций для всех заболеваний. В данном примере схемы правил представляют собой модифицированные правила третьего решателя задачи медицинской диагностики, в которых все префиксы убраны, а их функции выполняют, с одной стороны, кванторные циклы, а с другой – конкретизации правил.

Для данного эксперимента был использован второй набор данных из эксперимента 1. Время работы решателя для разного числа процессоров в испытании для второго набора данных приведено в таблице 8.

Табл. 8. Время работы приложения в ходе эксперимента 3 на втором наборе данных из эксперимента 1

Число процессоров	**1**	**2**	**4**	**6**	**9**	**29**	**39**
модуль 1	27,2 сек	14,5 сек	7,6 сек	5,6 сек	4,3 сек	3,1 сек	3,1 сек

При последовательном выполнении модулей (число рабочих процессоров равно 1) время работы составило 27,2 секунды. При параллельном режиме при увеличении числа процессоров время работы модулей уменьшилось и при 29 процессорах составило 3,1 секунды. По сравнению с последовательным режимом уменьшение - в 8,7 раза на 29 рабочих процессорах.

Сравним результаты эксперимента 1 на втором наборе данных и эксперимента 3. Время работы 1 решателя по сравнению со временем работы 3 решателя для последовательного режима совпадает, для параллельного режима при 29 процессорах уменьшилось в 2,2 раза. Уменьшение времени вычислений связано с отсутствием префиксов у правил, то есть с сокращением накладных расходов на запуск процессов.

Эксперимент 4. Экспертная система медицинской диагностики заболеваний острого живота

В рамках данного эксперимента была использована экспертная система (ЭС) Консультант-2, предназначенная для диагностики ряда острых хирургических заболеваний органов брюшной полости. Формальное представление базы знаний в виде семантической сети для этой системы было разработано М.Ю. Черняховской [11]. Для решения задачи медицинской диагностики используется метод опровержения гипотез.

Экспертная система состоит из 60 логических модулей, которые в сумме содержат 1978 правил опровержения (правила были построены автоматически компилятором правил по базе знаний [13,14]. Данная экспертная система решает ту же самую задачу, что и решатели задачи медицинской диагностики, описанные в экспериментах 1, 2 и 3. В отличие от

предыдущих экспериментов, в которых знания о заболеваниях хранятся в базе данных, в данном эксперименте знания "встроены" в правила.

В первом испытании системы при последовательном выполнении модулей время работы составило 101 секунду. В параллельном режиме, когда каждому модулю было выделено 10 рабочих процессоров, время работы составило 258 секунд. Увеличение времени работы связано с тем, что все правила в модулях являются простыми, независимыми друг от друга, время обработки каждого правила не превышает десятые доли секунды. Большую часть времени составили системные задержки по транспортировке данных между процессами.

В следующем эксперименте для параллельного выполнения все логические модули были разбиты на две группы по зависимым входным данным. В первую группу вошло 39 модулей, во вторую – 21. Для каждой из данных групп правил был сформирован управляющий файл, исполнение которого позволило запустить модули независимо друг от друга. Правила в каждом модуле работали последовательно. Во втором испытании при параллельном выполнении модулей на кластере время работы всего приложения составило 11 секунд. В параллельном режиме эффективность вычислений возросла в 9,1 раза по сравнению с последовательным режимом. Из результатов экспериментов видно, что уменьшение числа операций, выполняемых правилами, привело к увеличению времени работы в параллельном режиме за счет увеличения соотношения накладных расходов на запуск параллельных процессов к времени выполнения отдельного правила.

Система решения задачи Дирихле для уравнений эллиптического типа

В ходе данного эксперимента было создано приложение, решающее численно задачу Дирихле для уравнения Лапласа:

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = 0;$$

$(x,y) \in D$; $u = xy^2 = f(x,y)$; область D ограничена линиями: x=2, x=4, y=x, y=x+4; $(x_0, y_0) = (3, 5)$; шаг по x и по y: h = 0.08. Точность решения СЛАУ $\varepsilon = 0.01$. При решении задачи использован сеточный метод.

В состав приложения входит 3 логических модуля, которые в сумме описывают 12 правил. Первый модуль состоит из 1 правила, второй – из 2, третий – из 9.

В ходе эксперимента изучались временные характеристики системы решения задачи. Время работы приложения для разного числа процессоров приведено в таблице 9.

Табл. 9. Время работы приложения для задачи Дирихле

Число процессоров	**1**	**2**	**4**	**6**	**7**	**9**	**29**
модуль 1	0,6 сек	0,6 сек	0,6 сек	0,6 сек	0,6 сек	0,6 сек	0,6 сек
модуль 2	0,5 сек	0,4 сек	0,5 сек	0,5 сек	0,5 сек	0,5 сек	0,5 сек
модуль 3	164,5 сек	122,1 сек	93,3 сек	90,4 сек	122,7 сек	122,8 сек	122,9 сек
Всего времени	165,6 сек	123,1 сек	94,4 сек	91,5 сек	123,8 сек	123,9 сек	124 сек

При последовательном выполнении модулей (число рабочих процессоров равно 1) время работы составило 165,6 секунды. При параллельном режиме при увеличении числа процессоров время работы модулей уменьшилось, при 6 процессорах время работы составило 91,5 секунды и уменьшилось по сравнению с последовательным режимом в 1,8 раза. Уменьшение времени связано с распараллеливанием логического вывода с использованием схемы распараллеливания по информационному графу. Однако при 7

процессорах и выше время увеличивается относительно 6 процессоров, так как в процессе логического вывода поменялся порядок обработки ветвей информационного графа логического модуля 3 из-за увеличения числа доступных процессов, что повлекло изменение порядка выполнения правил относительно друг друга. Следствием этого стало уменьшение числа правил, обрабатываемых одновременно в параллельном режиме.

Задача поиска путей синтеза химических соединений

В ходе данного эксперимента был создан решатель одного из классов задач поиска путей синтеза химических соединений: задано множество химических реакций, их реагентов и результатов, а также соединение, которое требуется синтезировать, и соединения, используемые на первом шаге синтеза, требуется построить все возможные последовательности реакций, которые позволят синтезировать указанное соединение при заданных условиях.

Приложение состоит из одного модуля, содержащего 8 правил. Набор входных данных для приложения содержит модельные данные о 221 химической реакции, их реагентах и результатах. Время работы приложения для разного числа процессоров приведено в таблице 10.

Табл. 10. Время решение задачи поиска путей синтеза химических соединений

Число процессоров	**1**	**2**	**4**	**6**	**9**
модуль 1	32,1 сек	24,7 сек	12,9 сек	13,1 сек	13,1 сек

При последовательном выполнении модулей (число рабочих процессоров равно 1) время работы составило 32,1 секунды. При параллельном режиме при увеличении числа процессоров время работы модулей уменьшилось и при 6 процессорах составило 13,1 секунды. По сравнению с последовательным режимом уменьшение - в 2,4 раза на 6 рабочих процессорах. Уменьшение времени работы связано с распараллеливанием вычислений с использованием схемы распараллеливания по информационному графу.

Выводы

В работе описаны экспериментальные исследования свойств системы на различных наборах данных, которые показывают эффективность примененных схем распараллеливания процесса выполнения правил и метода управления выбором схем. В результате исследований было проведено 27 экспериментов, из 18 показывают работу приложений с разным строением информационного графа, 9 – показывают работу приложений для реальных предметных областей.

Эксперименты показали, что на большинстве наборов данных время работы приложений уменьшается при увеличении числа процессоров. Особенно эффективно работают приложения, для которых доступно большое количество входных данных, и которые позволяют существенно загрузить работой отдельные процессы (т.е. когда время выполнения правила больше времени, затрачиваемого на пересылку данных для него), причем лучшее время получено, когда число выделенных процессоров примерно равно числу параллельно запускаемых процессов. Потери времени для параллельного режима происходят, когда отдельное правило выполняется быстрее, чем формируются и пересылаются данные для него.

Список литературы

1. Артемьева И.Л., Клещев А.С. Необогащенные системы логических соотношений. Часть 1. // НТИ, сер. 2.- 2000.- № 7.- С. 18-28.
2. Артемьева И.Л., Клещев А.С. Необогащенные системы логических соотношений. Часть 2. // НТИ, сер. 2.- 2000.- № 8.- С. 8-18.
3. Артемьева И.Л., Тютюнник М.Б. Модульная система конфлюэнтных продукций для многопроцессорной вычислительной системы. // Программные продукты и системы. - 2007. - №2. - С. 38-39.
4. Артемьева И.Л., Тютюнник М.Б. Управление распараллеливанием логического вывода для системы, основанной на правилах // Научно-технические ведомости СПбГПУ.- 2008. - №3. - С.99-103.
5. Артемьева И.Л., Тютюнник М.Б. Модель модульного языка декларативных конфлюэнтных продукций с ограниченными кванторами. // Международная научно-техническая конференция "Искусственный интеллект - 2007. Интеллектуальные системы", 24-29 сентября 2007.
6. Artemieva I.L., Tyutyunnik M.B. Parallelization of Logical inference for confluent rule-based system // International Book Series "Information Science and Computing". Book 1 "Algorithmic and Mathematical Foundations of the Artificial Intelligence ". 2008. PP.81-87.
7. Артемьева И.Л., Тютюнник М.Б. Методы управлением логического вывода для продукционной систем // Управление создание и развитием систем, сетей и устройств телекоммуникаций / Под ред. д.эк.н., проф. А.В.Бабкина, д.т.н., проф. В.А.Кежаева: Тр. научн.-пр. конф., СПб.: НОЦ "Перспектива". 2008. – С. 60-63. - ISBN 5-7422-1583-5.
8. Артемьева И.Л.. Тютюнник М.Б. Исследование распределенной системы логического программирования. Труды IV Международной конференции «Параллельные вычисления и задачи управления» PACO'2008. Москва. Институт проблем управления им. В.А. Трапезникова РАН. М.: Институт проблем управления им. В.А. Трапезникова РАН, 2008.
9. Вагин В.Н., Головина Е.Ю., Загорянская А.А., Фомина М.В. Достоверный вывод в интеллектуальных системах / Под ред. В.Н. Вагина, Д.А. Поспелова. – М.: ФИЗМАТЛИТ, 2004. – 704 с.
10. Клещев А.С. Реляционная модель вычислений // Программирование, 1980, 4. С. 20-29.
11. Черняховская М.Ю. Представление знаний в экспертных системах медицинской диагностики. // Владивосток, ДВНЦ АН СССР, 1983. 212 с.
12. Яхно Т.М. Системы продукций: структура, технология, применение. Новосибирск: ВЦ СО АН СССР. 1990. 127 с.
13. Клещев А.С., Самсонов В.В. МАКРОРЕПРО – язык спецификации компиляции баз знаний в базы правил: Препринт. Владивосток: ДВО РАН, 1992. 25 с.
14. Самсонов В.В., Сорокин В.С., Черняховская М.Ю. Экспериментальная медицинская экспертная система Консультант-2 // Проблемы проектирования экспертных систем: Тез. Докл. Всесоюз. школы-совещания. Ч.2 Москва, 1988. С. 238-239.
15. http://www.w3.org/TR/rif-ucr/