Применение предикции при параллельной обработке цепочек предикатов в регулярно-логических выражениях

**Человек:** В данной работе рассматривается проблема выбора режима исполнения (последовательный или параллельный) при обработке цепочек предикатов в регулярно-логических выражениях. Дано краткое описание сути регулярно-логических выражений, их известных применений (естественно-языковые интерфейсы, автоматический параллелизатор C-программ), видов и состава цепочек предикатов. Особое внимание уделяется вопросу предикции временных затрат при обработке цепочек в том или ином режиме. Подробно рассматриваются различные подходы к такой возможной предикции. Отмечено, что в данном случае наиболее естественнен полуэмпирико-статистический подход. В работе используются основные соотношения теории параллельных вычислений, методы интерполяции и экстраполяции, вычислительного эксперимента, элементы статистической обработки. Предлагается новый полуэмпирико-статистический подход к решению проблемы вычисления оценок времени исполнения цепочек предикатов. Подход отличается минимальным количеством замеров времени, достигнутым с помощью частичного восстановления недостающих данных, и применением потенциально более точных линейных авторегрессионных и квадратичных моделей для вычисления предполагаемого времени исполнения в последовательном или параллельном режимах.

**Key words:** регулярно-логические выражения, логическая обработка, цепочки предикатов, выбор режима исполнения, параллельная обработка, предикция, сбор статистики, восстановление пропущенных данных, полуэмпирические оценки, вычислительный эксперимент

=================================

**FastText\_KMeans\_Clean:** Применение аналитических соотношений [11, 12, 13] – малозатратный подход, который может быть достаточно совершенным с точки зрения предсказания, но в нашем случае построение таких универсальных моделей времени для произвольных предикатов, в том числе имеющих дело с не всегда упорядоченными данными в CSV-таблицах и нейронными сетями, достаточно затруднительно. Данная проблема разрешима:. После серии начальных замеров включаем рабочий режим (на некоторое количество P вычислений цепочки, после которого делаем сброс статистики с переходом к пункту 1), при котором для p-го вычисления цепочки (p > K) делается прогноз значения t1,p, по которому вычисляется прогноз tN,p. б) использование линейного предиктора времени исполнения последовательного режима и квадратичной зависимости для времени исполнения параллельного режима;.

**Key words part:** 0.5333333333333333

=================================

**FastText\_KMeans\_Raw/:** Применение аналитических соотношений [11, 12, 13] – малозатратный подход, который может быть достаточно совершенным с точки зрения предсказания, но в нашем случае построение таких универсальных моделей времени для произвольных предикатов, в том числе имеющих дело с не всегда упорядоченными данными в CSV-таблицах и нейронными сетями, достаточно затруднительно. Формальная постановка задачи. Данная проблема разрешима:. 1. После каждого очередного сброса статистики можно ограничиться лишь одной небольшой серией из K вычислений цепочки (попеременно, в последовательном и параллельном режимах – с вариацией, например, через 1÷2 шага), в результате чего имеем серию замеров (при единичном шаге вариации). Полученные результаты сведены в таблицу.

**Key words part:** 0.5

=================================

**FastText\_PageRank\_Clean/:** Схожий принцип использован в работе [17]. Формальная постановка задачи. Предикция времени исполнения. Данная проблема разрешима:. Полученные значения t1 и tN сохраняются. Полученные результаты сведены в таблицу. Результаты проведенных экспериментов (замеры времени, мс). Ключевыми новыми элементами предложенной методики являются:.

**Key words part:** 0.43333333333333335

=================================

**FastText\_PageRank\_Raw/:** Схожий принцип использован в работе [17]. Формальная постановка задачи. Предикция времени исполнения. Данная проблема разрешима:. Полученные результаты сведены в таблицу. Результаты проведенных экспериментов (замеры времени, мс). б) несколько хуже, чем в априорно неизвестном лучшем случае. Ключевыми новыми элементами предложенной методики являются:.

**Key words part:** 0.43333333333333335

=================================

**Mixed\_ML\_TR/:** Применение аналитических соотношений [11, 12, 13] – малозатратный подход, который может быть достаточно совершенным с точки зрения предсказания, но в нашем случае построение таких универсальных моделей времени для произвольных предикатов, в том числе имеющих дело с не всегда упорядоченными данными в CSV-таблицах и нейронными сетями, достаточно затруднительно. В параллельном режиме можно совместить во времени исполнение независимых друг от друга предикатов, при этом можно организовать обработку по типу "портфель задач" [18], где одна задача соответствует одному предикату A и имеет в качестве предусловия выполнения завершение обработки всех предикатов, от которых зависит A. Остается решить задачу выбора режима исполнения. Предположим, что существует подход, позволяющий вычислить предполагаемое время исполнения в последовательном режиме t1 и в параллельном режиме tN. Однако такой подход не вполне эффективен, поскольку построение статистики для двух независимых экстраполяторов требует проведения двойной серии вычислений цепочки (одной в последовательном и одной в параллельном режиме), что, скорее всего, даст негативный эффект по времени исполнения. Данная проблема разрешима:. Из приведенных данных очевидно, что ни строго последовательный, ни строго параллельный режим исполнения не способны гарантировать наименьшее время исполнения во всех случаях (для трудоемких примеров, в которых доля полезной работы заметно выше затрат на распараллеливание, более выгоден параллельный режим, а для "легковесных", где ситуация обратная, – последовательный режим). Данная методика позволяет оперативно выбрать близкий к оптимальному режим исполнения (последовательный или параллельный), который гарантированно дает время исполнения:. в) применение периодической подстройки коэффициентов указанных предикционных выражений путем выполнения одного шага численного решения задачи минимизации квадратичной ошибки.

**Key words part:** 0.6666666666666666

=================================

**MultiLingual\_KMeans/:** Применение аналитических соотношений [11, 12, 13] – малозатратный подход, который может быть достаточно совершенным с точки зрения предсказания, но в нашем случае построение таких универсальных моделей времени для произвольных предикатов, в том числе имеющих дело с не всегда упорядоченными данными в CSV-таблицах и нейронными сетями, достаточно затруднительно. В параллельном режиме можно совместить во времени исполнение независимых друг от друга предикатов, при этом можно организовать обработку по типу "портфель задач" [18], где одна задача соответствует одному предикату A и имеет в качестве предусловия выполнения завершение обработки всех предикатов, от которых зависит A. Остается решить задачу выбора режима исполнения. Данная проблема разрешима:. в) применение периодической подстройки коэффициентов указанных предикционных выражений путем выполнения одного шага численного решения задачи минимизации квадратичной ошибки.

**Key words part:** 0.6

=================================

**Multilingual\_PageRank/:** Схожий принцип использован в работе [17]. Такой подход предлагается в данной работе. Формальная постановка задачи. Полученные значения t1 и tN сохраняются. б) "легковесной" цепочке из 40 независимых табличных предикатов, вызываемой 64 раза. Полученные результаты сведены в таблицу. б) несколько хуже, чем в априорно неизвестном лучшем случае. Ключевыми новыми элементами предложенной методики являются:.

**Key words part:** 0.4

=================================

**RuBERT\_KMeans\_Without\_ST/:** Данная проблема может быть решена путем предикции временных затрат как на параллельную, так и на последовательную обработку микровыражений с определением наиболее подходящего режима работы. Применение аналитических соотношений [11, 12, 13] – малозатратный подход, который может быть достаточно совершенным с точки зрения предсказания, но в нашем случае построение таких универсальных моделей времени для произвольных предикатов, в том числе имеющих дело с не всегда упорядоченными данными в CSV-таблицах и нейронными сетями, достаточно затруднительно. б) "легковесной" цепочке из 40 независимых табличных предикатов, вызываемой 64 раза. Из приведенных данных очевидно, что ни строго последовательный, ни строго параллельный режим исполнения не способны гарантировать наименьшее время исполнения во всех случаях (для трудоемких примеров, в которых доля полезной работы заметно выше затрат на распараллеливание, более выгоден параллельный режим, а для "легковесных", где ситуация обратная, – последовательный режим).

**Key words part:** 0.5666666666666667

=================================

**RuBERT\_KMeans\_With\_ST/:** Применение аналитических соотношений [11, 12, 13] – малозатратный подход, который может быть достаточно совершенным с точки зрения предсказания, но в нашем случае построение таких универсальных моделей времени для произвольных предикатов, в том числе имеющих дело с не всегда упорядоченными данными в CSV-таблицах и нейронными сетями, достаточно затруднительно. Более того, при таком подходе сомнения вызывают даже эффективность и гладкость параметризации решаемой нами задачи, учитывая, что большинство ее переменных связаны с поиском по полям постоянно модифицируемых CSV-таблиц. Предикция времени исполнения. Это позволяет говорить о достаточной обоснованности предложенных в данной работе подходов.

**Key words part:** 0.43333333333333335

=================================

**RUBERT\_page\_rank\_Without\_ST/:** Формальная постановка задачи. Предикция времени исполнения. где α – доля работы, не поддающейся распараллеливанию, β – коэффициент накладных расходов на распараллеливание. Полученные результаты сведены в таблицу. Результаты проведенных экспериментов (замеры времени, мс).

**Key words part:** 0.4

=================================

**RUBERT\_page\_rank\_With\_ST/:** Схожий принцип использован в работе [17]. Формальная постановка задачи. Данная проблема разрешима:. Полученные результаты сведены в таблицу. Результаты проведенных экспериментов (замеры времени, мс).

**Key words part:** 0.36666666666666653

=================================

**RUSBERT\_KMeans\_Without\_ST/:** Тот же самый эффект вполне вероятен для подхода [10], моделирующего логику работы кода (в соответствии с графом исполнения) с вычислением длительности счета отдельных фрагментов с помощью имитационной модели. Такой подход предлагается в данной работе. Предикция времени исполнения. Введя предположения о линейной зависимости α и β от t1, получаем. На протяжении рабочего режима эти значения могут использоваться для периодического пересчета коэффициентов предиктора, а также для небольшой постоянной коррекции этих коэффициентов, например, с помощью одного шага градиентного метода [19], исходя из принципа минимизации соответствующего функционала квадратичной ошибки. а) частичное восстановление недостающих статистичеких данных (по простым аналитическим соотношениям, без проведения дублирующих экспериментов);.

**Key words part:** 0.5

=================================

**RUSBERT\_KMeans\_With\_ST/:** Однако такой подход не вполне эффективен, поскольку построение статистики для двух независимых экстраполяторов требует проведения двойной серии вычислений цепочки (одной в последовательном и одной в параллельном режиме), что, скорее всего, даст негативный эффект по времени исполнения. Данная проблема разрешима:. На протяжении рабочего режима эти значения могут использоваться для периодического пересчета коэффициентов предиктора, а также для небольшой постоянной коррекции этих коэффициентов, например, с помощью одного шага градиентного метода [19], исходя из принципа минимизации соответствующего функционала квадратичной ошибки. Она, обычно, как очевидно из таблицы, не позволяет получить результаты с наименьшим временем (поскольку имеют место дополнительные временные затраты на собственно предикцию, также возможны ошибки в прогнозировании лучшего режима), однако стабильно дает приемлемое время счета, которое несколько не достигает лучшего варианта, но гарантированно лучше, чем худший вариант, который всегда может иметь место при априорном неправильном выборе одного из "чистых" режимов.

**Key words part:** 0.5666666666666667

=================================

**RUSBERT\_page\_rank\_Without\_ST/:** Схожий принцип использован в работе [17]. Такой подход предлагается в данной работе. Данная проблема разрешима:. Полученные значения t1 и tN сохраняются. б) несколько хуже, чем в априорно неизвестном лучшем случае.

**Key words part:** 0.3333333333333333

=================================

**RUSBERT\_page\_rank\_With\_ST/:** Схожий принцип использован в работе [17]. Такой подход предлагается в данной работе. Формальная постановка задачи. Полученные значения t1 и tN сохраняются. б) несколько хуже, чем в априорно неизвестном лучшем случае.

**Key words part:** 0.3333333333333333

=================================

**Simple\_PageRank/:** Микровыражения работают с именованными парами скобок (переменными) и могут иметь (для вида цепочки "проверка") логический результат, обозначающий успешность или неуспех (с откатом) попытки унификации микровыражения по аналогии с любым иным синтаксическим элементом регулярного выражения. Возможно возникновение ситуаций, при которых отдельные регулярно-логические выражения применяются неоднократно (например, при использовании сканирующей схемы индукции [4]), причем некоторые из предикатов, работающих с большими таблицами или решающих нейросетевые задачи, имеют высокую трудоемкость. Применение аналитических соотношений [11, 12, 13] – малозатратный подход, который может быть достаточно совершенным с точки зрения предсказания, но в нашем случае построение таких универсальных моделей времени для произвольных предикатов, в том числе имеющих дело с не всегда упорядоченными данными в CSV-таблицах и нейронными сетями, достаточно затруднительно. В параллельном режиме можно совместить во времени исполнение независимых друг от друга предикатов, при этом можно организовать обработку по типу "портфель задач" [18], где одна задача соответствует одному предикату A и имеет в качестве предусловия выполнения завершение обработки всех предикатов, от которых зависит A. Остается решить задачу выбора режима исполнения. Очевидным решением проблемы предикции, теоретически, могло бы стать, например, построение двух независимых экстраполяторов (первый – для t1, второй – для tN) по некоторой собираемой статистике, которую целесообразно периодически "сбрасывать" и собирать заново, для повышения валидности предикции. Она, обычно, как очевидно из таблицы, не позволяет получить результаты с наименьшим временем (поскольку имеют место дополнительные временные затраты на собственно предикцию, также возможны ошибки в прогнозировании лучшего режима), однако стабильно дает приемлемое время счета, которое несколько не достигает лучшего варианта, но гарантированно лучше, чем худший вариант, который всегда может иметь место при априорном неправильном выборе одного из "чистых" режимов.

**Key words part:** 0.7666666666666667

=================================

**TextRank/:** Применение аналитических соотношений [11, 12, 13] – малозатратный подход, который может быть достаточно совершенным с точки зрения предсказания, но в нашем случае построение таких универсальных моделей времени для произвольных предикатов, в том числе имеющих дело с не всегда упорядоченными данными в CSV-таблицах и нейронными сетями, достаточно затруднительно. В параллельном режиме можно совместить во времени исполнение независимых друг от друга предикатов, при этом можно организовать обработку по типу "портфель задач" [18], где одна задача соответствует одному предикату A и имеет в качестве предусловия выполнения завершение обработки всех предикатов, от которых зависит A. Остается решить задачу выбора режима исполнения. Предположим, что существует подход, позволяющий вычислить предполагаемое время исполнения в последовательном режиме t1 и в параллельном режиме tN. Однако такой подход не вполне эффективен, поскольку построение статистики для двух независимых экстраполяторов требует проведения двойной серии вычислений цепочки (одной в последовательном и одной в параллельном режиме), что, скорее всего, даст негативный эффект по времени исполнения. После серии начальных замеров включаем рабочий режим (на некоторое количество P вычислений цепочки, после которого делаем сброс статистики с переходом к пункту 1), при котором для p-го вычисления цепочки (p > K) делается прогноз значения t1,p, по которому вычисляется прогноз tN,p. Из приведенных данных очевидно, что ни строго последовательный, ни строго параллельный режим исполнения не способны гарантировать наименьшее время исполнения во всех случаях (для трудоемких примеров, в которых доля полезной работы заметно выше затрат на распараллеливание, более выгоден параллельный режим, а для "легковесных", где ситуация обратная, – последовательный режим).

**Key words part:** 0.6333333333333333

=================================

**TF-IDF\_KMeans/:** Регулярно-логические выражения представляют собой расширение обычных регулярных Perl-выражений [1], которые дополняются логическими микровыражениями – цепочками предикатов, записанных в нотации некоторого обобщенного логического языка. Данная проблема может быть решена путем предикции временных затрат как на параллельную, так и на последовательную обработку микровыражений с определением наиболее подходящего режима работы. Применение аналитических соотношений [11, 12, 13] – малозатратный подход, который может быть достаточно совершенным с точки зрения предсказания, но в нашем случае построение таких универсальных моделей времени для произвольных предикатов, в том числе имеющих дело с не всегда упорядоченными данными в CSV-таблицах и нейронными сетями, достаточно затруднительно. Полученные значения t1 и tN сохраняются. На протяжении рабочего режима эти значения могут использоваться для периодического пересчета коэффициентов предиктора, а также для небольшой постоянной коррекции этих коэффициентов, например, с помощью одного шага градиентного метода [19], исходя из принципа минимизации соответствующего функционала квадратичной ошибки. б) "легковесной" цепочке из 40 независимых табличных предикатов, вызываемой 64 раза. Она, обычно, как очевидно из таблицы, не позволяет получить результаты с наименьшим временем (поскольку имеют место дополнительные временные затраты на собственно предикцию, также возможны ошибки в прогнозировании лучшего режима), однако стабильно дает приемлемое время счета, которое несколько не достигает лучшего варианта, но гарантированно лучше, чем худший вариант, который всегда может иметь место при априорном неправильном выборе одного из "чистых" режимов. б) использование линейного предиктора времени исполнения последовательного режима и квадратичной зависимости для времени исполнения параллельного режима;.

**Key words part:** 0.7333333333333333

=================================

**Текст:** Регулярно-логические выражения представляют собой расширение обычных регулярных Perl-выражений [1], которые дополняются логическими микровыражениями – цепочками предикатов, записанных в нотации некоторого обобщенного логического языка. Существует четыре вида таких цепочек: добавление, удаление, проверка и обучение. При этом выделяются следующие классы предикатов: а) стандартные табличные, осуществляющие работу с CSV-таблицей (четкий или нечеткий поиск, добавление или удаление текстовых или числовых значений), б) стандартные нейросетевые, поддерживающие создание и обучение нейросети, классификацию сетью текстовых или числовых значений, а также восстановление сетью некоторых отсутствующих классифицирующих признаков по известному классу, в) произвольные внешние предикаты, подключаемые из динамических библиотек. Микровыражения работают с именованными парами скобок (переменными) и могут иметь (для вида цепочки «проверка») логический результат, обозначающий успешность или неуспех (с откатом) попытки унификации микровыражения по аналогии с любым иным синтаксическим элементом регулярного выражения.. Регулярно-логические выражения используются в индуцирующих классах распознающих моделей в системе порождения и реконструкции программ PGEN++ [2, 3]. С их помощью реализуются элементы естественно-языкового интерфейса данной системы [4, 5], а также макромодуль, решающий такую достаточно актуальную задачу как автоматическое распараллеливание C-программ в Cilk++ стиле [4].. Возможно возникновение ситуаций, при которых отдельные регулярно-логические выражения применяются неоднократно (например, при использовании сканирующей схемы индукции [4]), причем некоторые из предикатов, работающих с большими таблицами или решающих нейросетевые задачи, имеют высокую трудоемкость. В таком случае актуально повышение производительности обработки микровыражений (цепочек предикатов). Данная цель может быть отчасти достигнута применением параллельной обработки таких микровыражений, например, для случая системы с общей памятью [6], к которому относится большинство современных ЭВМ с многоядерными процессорами.. К сожалению, время исполнения отдельных предикатов цепочки может существенно варьироваться от запуска к запуску, поэтому распараллеливание не всегда дает положительный эффект, иногда оно приводит даже к повышению временных затрат (это вполне возможно, если дополнительные временные затраты на распараллеливание превысили положительный эффект от такого распараллеливания). Данная проблема может быть решена путем предикции временных затрат как на параллельную, так и на последовательную обработку микровыражений с определением наиболее подходящего режима работы.. Существуют три основных подхода к предикции: а) имитационное моделирование; б) вывод и применение аналитических соотношений; в) применение полуэмпирических соотношений на базе некоторой известной статистики. Следует заметить, что имитационное моделирование [7, 8, 9] – достаточно тяжеловесный подход, затраты времени на который могут превысить эффект от распараллеливания. Особенно это может быть характерно для подхода с прямым моделированием, основанным на частичном фактическом исполнении исходного кода, упомянутого в работе [9]. Тот же самый эффект вполне вероятен для подхода [10], моделирующего логику работы кода (в соответствии с графом исполнения) с вычислением длительности счета отдельных фрагментов с помощью имитационной модели. Применение аналитических соотношений [11, 12, 13] – малозатратный подход, который может быть достаточно совершенным с точки зрения предсказания, но в нашем случае построение таких универсальных моделей времени для произвольных предикатов, в том числе имеющих дело с не всегда упорядоченными данными в CSV-таблицах и нейронными сетями, достаточно затруднительно. Решением может быть применение полуэмпирико-статистических соотношений. Однако такие известные решения [7, 14] опираются на достаточно простые линейные соотношения и требуют априорно большого числа замеров. Первое говорит об их потенциальной неточности, второе - об их потенциальной неэффективности по времени, что может быть особенно характерно для рассматриваемого нами общего случая, когда некоторые микровыражения имеют вычислительную трудоемкость, меньшую трудоемкости такой предикции. Достаточно интересен подход, изложенный в работах [15, 16], основанный на сборе статистики, классифицируемой по значениям параметров решаемой задачи, и ее применении в линейных моделях времени исполнения. Схожий принцип использован в работе [17]. Одним из недостатков этого подхода является потенциально большой объем хранимой и используемой в вычислениях статистики. Более того, при таком подходе сомнения вызывают даже эффективность и гладкость параметризации решаемой нами задачи, учитывая, что большинство ее переменных связаны с поиском по полям постоянно модифицируемых CSV-таблиц. В таких условиях параметризация, скорее, неэффективна и можно полагаться лишь на часть последних по времени замеров, что говорит о возможности применения авторегрессионных моделей. Приходим к выводу, что для рассматриваемой задачи актуален поиск нового полуэмпирико-статистического подхода, отличающегося минимальным количеством замеров времени и применением потенциально более точных моделей (линейных авторегрессионных и квадратичных моделей времени исполнения). Такой подход предлагается в данной работе.. . Формальная постановка задачи. Пусть имеется цепочка предикатов (микровыражение), которая многократно вычисляется в ходе унификации регулярно-логического выражения, причем время исполнения цепочки априорно неизвестно. Часть предикатов цепочки зависимы по данным от некоторых предыдущих ее предикатов. В последовательном режиме цепочка вычисляется строго слева направо. В параллельном режиме можно совместить во времени исполнение независимых друг от друга предикатов, при этом можно организовать обработку по типу «портфель задач» [18], где одна задача соответствует одному предикату A и имеет в качестве предусловия выполнения завершение обработки всех предикатов, от которых зависит A.. Остается решить задачу выбора режима исполнения. Предположим, что существует подход, позволяющий вычислить предполагаемое время исполнения в последовательном режиме t1 и в параллельном режиме tN. Тогда, при t1 ≤ tN выбирается последовательный режим, иначе – параллельный.. . Предикция времени исполнения. Очевидным решением проблемы предикции, теоретически, могло бы стать, например, построение двух независимых экстраполяторов (первый – для t1, второй – для tN) по некоторой собираемой статистике, которую целесообразно периодически «сбрасывать» и собирать заново, для повышения валидности предикции. Однако такой подход не вполне эффективен, поскольку построение статистики для двух независимых экстраполяторов требует проведения двойной серии вычислений цепочки (одной в последовательном и одной в параллельном режиме), что, скорее всего, даст негативный эффект по времени исполнения.. Данная проблема разрешима:. 1. После каждого очередного сброса статистики можно ограничиться лишь одной небольшой серией из K вычислений цепочки (попеременно, в последовательном и параллельном режимах – с вариацией, например, через 1÷2 шага), в результате чего имеем серию замеров (при единичном шаге вариации). . по которой можно интерполировать (в простейшем случае – путем линейной интерполяции) недостающие значения. . и. .. 2. Учитывая очевидное существование связи между t1 и tN, можно ограничиваться построением одного линейного предиктора только для t1:. ,. где S – номер замера времени, a – вектор коэффициентов предиктора, легко вычисляемый по серии из K предыдущих замеров, K > 3. Что же касается вычисления tN, то можно ограничиться гипотетическим соотношением (полученным, исходя из известных соотношений [6]):. ,. где α – доля работы, не поддающейся распараллеливанию, β – коэффициент накладных расходов на распараллеливание. Введя предположения о линейной зависимости α и β от t1, получаем. ,. где m, p, q – неизвестные коэффициенты, которые можно вычислить по серии из K замеров t1 и tN, воспользовавшись методом наименьших квадратов [19]. Важно отметить, что последнее соотношение также позволяет вычислить t1 по известному значению tN, найдя максимальный из вещественных корней соответствующего квадратного уравнения (если вещественных корней нет, можно, с известной погрешностью, брать предыдущее значение t1).. 3. После получения указанных в пункте 1 серий из K значений, можно вычислить (см. пункт 2) все неизвестные коэффициенты, связанные с предикцией: a, m, p, q.. 4. После серии начальных замеров включаем рабочий режим (на некоторое количество P вычислений цепочки, после которого делаем сброс статистики с переходом к пункту 1), при котором для p-го вычисления цепочки (p > K) делается прогноз значения t1,p, по которому вычисляется прогноз tN,p. Если t1,p ≤ tN,p, то p-й запуск проходит в последовательном режиме (при этом получаем реальное значение t1, по которому восстанавливаем предполагаемое tN), иначе – в параллельном режиме (при этом получаем реальное значение tN, по которому восстанавливаем предполагаемое t1). Полученные значения t1 и tN сохраняются. На протяжении рабочего режима эти значения могут использоваться для периодического пересчета коэффициентов предиктора, а также для небольшой постоянной коррекции этих коэффициентов, например, с помощью одного шага градиентного метода [19], исходя из принципа минимизации соответствующего функционала квадратичной ошибки. . ` `. или. .. . Апробация. Предложенная методика выбора режима исполнения была опробована на двух характерных примерах:. а) трудоемкой цепочке из 6 предикатов (как табличных, так и нейросетевых) с зависимостями, вызываемой 32 раза;. б) «легковесной» цепочке из 40 независимых табличных предикатов, вызываемой 64 раза.. Каждый пример исполнялся в двух сериях, в каждой из которых испытывались три режима исполнения (последовательный, параллельный, с предикцией). Полученные результаты сведены в таблицу.. . Таблица. Результаты проведенных экспериментов (замеры времени, мс). Пример. Номер серии. Режим исполнения. Последовательный. Параллельный. С предикцией. Трудоемкий. 1. 24579. 20829. 21959. 2. 25470. 21671. 21986. «Легковесный». 1. 1265. 2079. 1515. 2. 1265. 2077. 1453. . Из приведенных данных очевидно, что ни строго последовательный, ни строго параллельный режим исполнения не способны гарантировать наименьшее время исполнения во всех случаях (для трудоемких примеров, в которых доля полезной работы заметно выше затрат на распараллеливание, более выгоден параллельный режим, а для «легковесных», где ситуация обратная, – последовательный режим). Более того, было бы крайне затруднительно, если вообще возможно (в общем случае), априорно автоматически выбирать из этих режимов лучший, поскольку такой обоснованный выбор потребует применения достаточно громоздких схем, например, решения задач оптимизации или имитационного моделирования, в том числе с анализом реализующего обработку цепочек кода.. Поэтому, априорно близким к оптимальному решением должна стать предложенная схема с предикцией, которая пытается выбрать лучший режим динамически. Она, обычно, как очевидно из таблицы, не позволяет получить результаты с наименьшим временем (поскольку имеют место дополнительные временные затраты на собственно предикцию, также возможны ошибки в прогнозировании лучшего режима), однако стабильно дает приемлемое время счета, которое несколько не достигает лучшего варианта, но гарантированно лучше, чем худший вариант, который всегда может иметь место при априорном неправильном выборе одного из «чистых» режимов. Это позволяет говорить о достаточной обоснованности предложенных в данной работе подходов.. . Выводы. Итак, в данной работе предложена новая методика предикции временных затрат при обработке цепочек предикатов в регулярно-логических выражениях. Данная методика позволяет оперативно выбрать близкий к оптимальному режим исполнения (последовательный или параллельный), который гарантированно дает время исполнения:. а) лучше, чем в худшем случае, который весьма вероятен при априорном выборе любого из «чистых» режимов исполнения;. б) несколько хуже, чем в априорно неизвестном лучшем случае.. Ключевыми новыми элементами предложенной методики являются:. а) частичное восстановление недостающих статистичеких данных (по простым аналитическим соотношениям, без проведения дублирующих экспериментов);. б) использование линейного предиктора времени исполнения последовательного режима и квадратичной зависимости для времени исполнения параллельного режима;. в) применение периодической подстройки коэффициентов указанных предикционных выражений путем выполнения одного шага численного решения задачи минимизации квадратичной ошибки.. Серия экспериментов, проведенная для цепочек предикатов различной трудоемкости, подтвердила заявленные оптимальные свойства методики: полученные результаты стабильно лучше худшего случая, хотя и не достигают результатов априорно неизвестного лучшего случая.