**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ**

**РОССИЙСКОЙ ФЕДЕРАЦИИ**

**ФИЛИАЛ ФЕДЕРАЛЬНОГО ГОСУДАРСТВЕННОГО БЮДЖЕТНОГО**

**ОБРАЗОВАТЕЛЬНОГО УЧРЕЖДЕНИЯ ВЫСШЕГО ОБРАЗОВАНИЯ**

**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ «МЭИ»**

**В Г. СМОЛЕНСКЕ**

Кафедра **«Вычислительная техника»**

Направление **09.04.01 «Информатика и вычислительная техника»**

магистерская программа **«Информационное и программное обеспечение автоматизированных систем»**

**ОТЧЁТ**

**по учебной практике**

студента **1** курса \_\_\_\_\_\_\_группы\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись) (фамилия, инициалы)

Место прохождение практики: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(указать место прохождения практики)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Отчёт сдан «\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_\_ г.

Руководитель практики от образовательной организации:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(должность) (подпись) (расшифровка подписи)

Защита отчёта состоялась «\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_20\_\_\_ г.

Оценка за практику \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(неудовлетворительно, удовлетворительно, хорошо, отлично)

Члены комиссии:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(должность) (подпись) (расшифровка подписи)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(должность) (подпись) (расшифровка подписи)

«\_\_\_»\_\_\_\_\_\_\_\_\_20\_\_г.

Смоленск 20\_\_\_

Оглавление

[Подготовительный этап 3](#_Toc473207310)

[Основной этап 4](#_Toc473207311)

[Заключительный этап 34](#_Toc473207312)

# 1 Подготовительный этап

Публикация «Topical Web crawlers: Evaluating adaptive algorithms» за авторством Филиппо Ментцера на сайте https://www.researchgate.net.

Профессор информатики; директор центра сложного исследования сетей и систем Университета Индианы.

Преподаватель информатики, профессор физики и участник программы когнитивистики в Университете Индианы, Блумингтон. Лауреат в области физики из Университета Рима и доктор Информатики и Когнитивистики Калифорнийского университета, Сан-Диего. Он в настоящее время является директором Центра Сложного Исследования Сетей , Старшим Научным сотрудником Института Kinsey. Ранее служил председателем подразделения в Школе IUB Информатики и Вычислений, и был Членом Института Санта-Фе. Он фокусируется на веб-науке, социальных сетях, средствах социального общения, социальном вычислении, интеллектуальных Веб-приложениях и моделировании сложных информационных сетей. Его работа была опубликована в Нью-Йорк Таймс, Wall Street Journal, NPR, CNN, USA Today, Washington Post, BBC, и многих других изданиях США и источниках международных новостей.

# 2 Основной этап

## 2.1 Перевод страниц публикации.

Актуальные поисковый роботы: Оценка адаптивных алгоритмов.

1. ВВЕДЕНИЕ

Поиск в Интернете является сложной задачей. Много машинного обучения прикладывается к одной из частей этой задачи, а именно ранжированию проиндексированных страниц по их оценкам по отношению к запросам пользователей. Это важная задача, поскольку она в значительной степени влияет на воспринимаемую эффективность поисковой системы. Пользователи часто смотрят лишь на несколько лучших ответов, в результате чего точность достигается путем ранжирования алгоритма первостепенной важности. Ранние поисковые системы выдавали страницы, главным образом схожие с лексической составляющей запроса. Ключевая стратегия состояла в том, чтобы разработать лучший алгоритм взвешивания для представления веб-страниц и запросов в векторном пространстве, таким образом, что близость в таком пространстве будет коррелироваться с семантической значимостью.

Совсем недавно структура гипертекстовых ссылок была признана в качестве мощного нового источника данных для веб-семантики. Многие методы машинного обучения были использованы для анализа линий связи, таким образом, что связь страницы была только на другие страницы, вместе с его содержимым, может быть использовано для оценки его актуальности. Наиболее известным примером такого анализа ссылок является алгоритм PageRank, он успешно используется в поисковой системе Google. Другие методы машинного обучения для извлечения значений из топологии связей основаны на идентификации страниц находящихся в центре внимания и авторитетных страниц с помощью собственных значений анализа и на других граф-теоретических подходах. В то время как ссылки на основе метода были признаны эффективными в некоторых случаях, анализ ссылки чаще всего в сочетании с лексическими до / после фильтрации.

Тем не менее, все это исследование касается только половины проблемы. Независимо от того, насколько сложный алгоритм ранжирования мы строим, результаты не могут быть столь же хороши, как для страницы проиндексированых поисковой системой - страница не может быть восстановлена, если она не была проиндексирована. Это подводит нас к другому аспекту веб-поиска, а именно сканированию веб-страниц в поисках страниц, которые будут индексироваться. Видимая Web с его приблизительным размером от 4 до 10 миллиардов "статических"страниц на момент написания кроет сложную проблему поиска информации. Освещение в Интернете с помощью поисковых систем не улучшилось за последние несколько лет. Даже с увеличением аппаратных средств и ресурсов полосы пропускания в их распоряжении, поисковые системы не могут идти в ногу с ростом Интернета. Поиск проблемы усугубляется тем, что веб-страницы также часто меняются. Несмотря на героические попытки поисковых систем индексировать весь Web, ожидается, что подпространство ускользает, индексация будет продолжать расти. Следовательно, решение, предложенное поисковыми системами, т.е. способность отвечать на любой запрос от любого пользователя распознается ограничено. Поэтому нет ничего удивительного в том, что разработка актуальных алгоритмов сканирования получил значительное внимание в последние годы.

Актуальные поисковые роботы реагируют на конкретные потребностей в информации, выраженных актуальных запросов или интересов профилей. Это могут быть потребности отдельного пользователя или общими интересами. Актуальные поисковые роботы поддерживают децентрализацию процесса сканирования, который является более масштабируемый подходом. Дополнительным преимуществом является то, что такие роботы могут управляться с богатым контекстом (темы, запросы, профили пользователей), внутри которого истолковываются страницы и выбраются ссылки для посещения.

Начиная с Пинкертона 1994 и De Bra и Post 1994 сканеров, определяющие начало исследований на сканировании, можно видеть множество алгоритмов. Существует Shark Поиск, более агрессивный вариант Fish Поиск De [1994]. Есть сканеры, чьи решения в значительной степени зависят от ссылок на основе критериев [Cho и др. 1998; Diligenti и др. 2000]. Diligenti и др. [2000], например, используют обратные ссылки на основе контекста графики, чтобы оценить вероятность страницы, ведущей к соответствующей странице, даже если исходная страница сама по себе не является актуальной. Другие используют лексические и концептуальные знания. Например, Чакрабарти и др. [1999] использовали иерархический классификатор выбора ссылок для обхода. Тем не менее другие подчеркивают контекстные знания [Aggarwal и др. 2001; Menczer и Белью 2000; Najork и Винер 2001] по этой теме в том числе, которые получили по релевантности обратной связи. Например Аггарваль [2001] узнает статистическую модель соответствующую особенностям для темы во время обхода. В предыдущей работе одного из авторов, Menczer и Бель[2000] показывают, что в хорошо организованной части Интернета, эффективные стратегии сканирование могут быть изучены и эволюционированы агентами с использованием нейронных сетей и эволюционных алгоритмов.

В настоящее время ( статья 2004 года ) идет творческая фаза касательно дизайна, сопровождается исследованиями по оценке таких сканеров, что является сложной проблемой. Например, при вызове к Веб-сканеру величина поисковых результатов ограничивает доступность пользователей на основе релевантности решений. В предыдущих исследованиях мы начали исследовать несколько альтернативных подходов как для оценки качества веб-страниц, а также для подведения итогов эффективности сканеров [Menczer и др. 2001]. В другой статье [Srinivasan и др. 2002] мы расширяли такую методологию, описывая подробно структуру, разработанную для оценки актуальных поисковых роботов. Анализ производительности основан на качестве и на использовании пространственных ресурсов. Мы формализуем класс сканирования задач возрастающей сложности, предлагаем ряд метрик оценки на основе различных доступных источников актуальности данных, анализируем временную сложность и масштабируемость ряда этих алгоритмов, предложенных в литературе, а также изучаем взаимосвязь между производительностью различных поисковых роботов и различных тематических характеристик.

Нашей целью в данной работе является изучение алгоритмических аспектов актуальных поисковых роботов. Мы реализовали в рамках нашей системы оценки групп алгоритмов, которые являются репрезентативными из доминирующих разновидностей, опубликованных в литературе. На основе их оценки в той или иной задачи, мы разработали и внедрили два новых класса сканер алгоритмов, которые в настоящее время являются самыми эффективными для выполнения задачи. Эти предлагаемые классы позволяют сосредоточиться на двух важных вопросах машинного обучения, которые не были ранее изучены в области веб-сканирования стратегий: (1) роль разведки против эксплуатации, и (2) роль адаптации (обучения и эволюционный алгоритм) в сравнении статических подходов. В целом, наше намерение состоит в том, чтобы получить более полную и достоверную картину относительных преимуществ и недостатков различных сканирующих стратегий.

В разделе 2 мы суммируем аспекты нашей системы оценки, которые имеют отношение к этому исследованию. Это включает в себя архитектуру системы, формализацию сканирования задачи, описание набора данных, показатели оценки, используемые для оценки эффективности. В разделе 3 описывается ряд алгоритмов, предложенных в литературе. Масштабируемость алгоритмов анализируется также путем изменения ресурсных ограничений по сканерам. Класс самых первых алгоритмов и класс алгоритмов SHARK -поиска введены в Разделе 4 и используется для изучения компромиссов между разведкой и эксплуатацией. Вопрос об адаптации рассматривается в разделе 5, с использованием класса многоагента алгоритмов, в которых люди могут научиться оценивать ссылки, укрепленные на основе локальных связи,. В разделе 6 мы анализируем надежности наших результатов в условиях длительных работ/использования, и, наконец, в Разделе 7 обсуждаются наши выводы и делаются вывод с идеями о дальнейших исследованиях.

2. EVALUATION FRAMEWORK(оценка)

Для того, чтобы опытным путем изучить алгоритмические проблемы, описанные выше, мы должны сначала проиллюстрировать структуру оценки, которые мы будем использовать в наших экспериментах. Здесь мы сосредоточимся только на аспектах рамок, которые имеют отношение к анализу в данной статье.

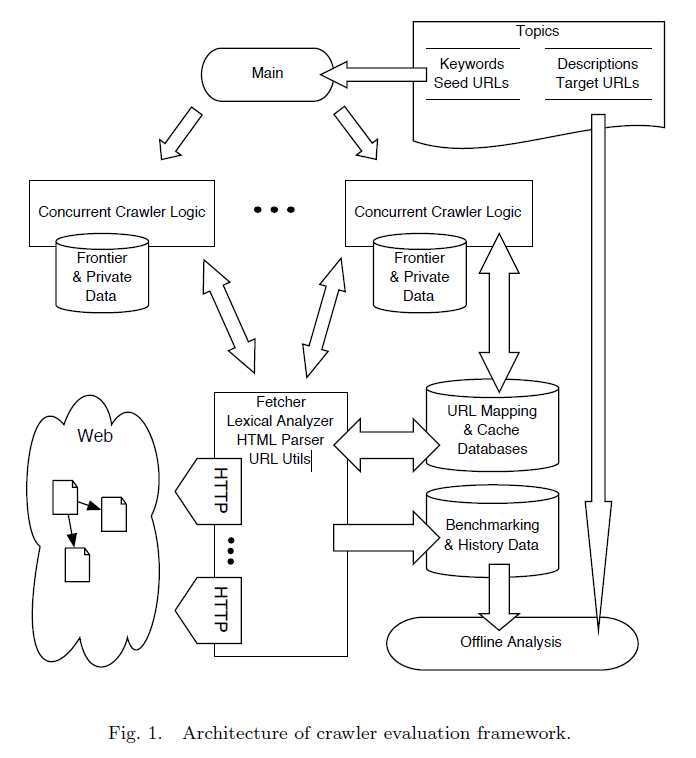
2.1 Архитектура системы

На рисунке 1 показана архитектура crawler evaluation framework. Система спроектирована таким образом, чтобы вся логика о каком-либо конкретном алгоритме инкапсулируется в модуле crawler(С), который может быть легко включен в систему через стандартный интерфейс. Все crawler модули с целью оптимизации производительности без ущерба для оценок. Примеры общих объектов включают в себя кэш-память, интерфейс HTTP для Web, простой HTML-анализатор, а Штеммер [Porter 1980], контрольные показатели и процедуры отчетности. Система реализована в Perl встроенных баз данных для хранения постоянных данных, а также структуры данных, общими для всех параллельных процессов.

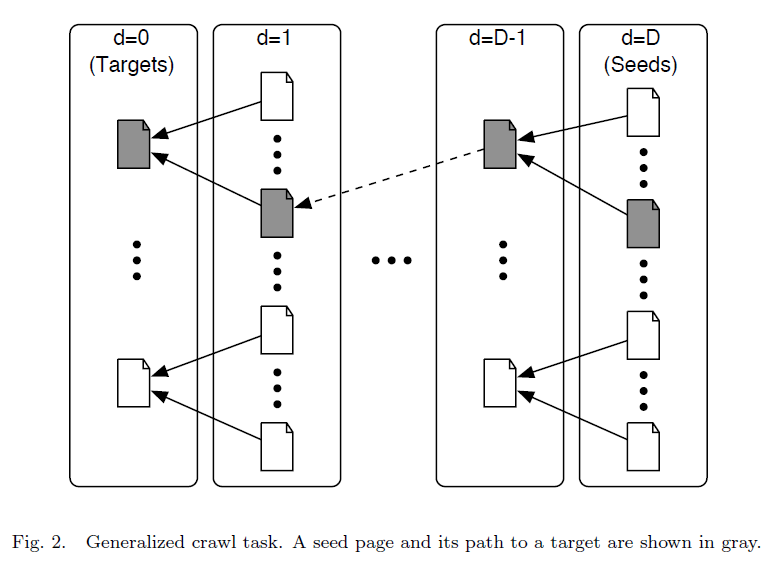
Каждый трестируемый crawler может посетить MAX PAGES. Мы используем время ожидания 10 секунд для загрузки веб страницы. Большие страницы расколоты/не целые, так что мы получаем только первые 10 Кбайт. Единственный протокол позволивший это HTTP GET (с переадресацией), и мы также отфильтровали все страницы с содержанием текста / HTML. Устаревшие ссылки, приносящие коды ошибок HTTP удаляются по мере их обнаружения; только хорошие ссылки используются в анализе.

2.2 Ограниченность ресурсов

С потребляют ресурсы: пропускная способность сети для загрузки страниц, память для поддержания частных структур данных в поддержку своих алгоритмов, CPU для оценки и выбора URL-адресов, и дисковое хранилище для



хранения текста и ссылки страниц, а также другие постоянные данные . Очевидно, что чем сложнее алгоритм выбора, тем больше использование таких ресурсов. Для того, чтобы обеспечить справедливое сравнения разнообразных С алгоритмов, мы принимаем две меры:

(1) Мы отслеживаем время процессора каждым С алгоритмом для каждой страницы и каждой теме, не обращая внимания на время, потраченное модулем извлечения, который является общим для всех С. Мы делаем это, так как невозможно контролировать сетевой трафик и заторы, и мы хотим  benchmark(глагол) операции только на С. Наблюдаемоепроцессорное время будет использоваться для сравнения сложностиС алгоритмов.

(2) Мы ограничиваем объем памяти, доступной для каждого С, ограничивая размер своего буфера(?). Этот буфер используется С для временного хранения данных линии связи, как правило, границу страниц, чьи ссылки не были изучены. Каждый С разрешается отслеживать максимум MAX BUFFER ссылки. Если буфер заполняется, С должен решить, какие ссылки должны быть заменены, когда добавляются новые. Значение параметра MAX BUFFER будет варьироваться от 28 до 211, чтобы оценитьпроизводительность С алгоритмов с их ресурсами памяти. Обратите внимание, что мы устанавливаем размер буфера для таких малых значений, потому что мы особенно заинтересованы в анализе производительности обходчика(?) в самых сложных условиях,то есть, когда памяти очень мало. Это было бы реально, например, если С был развернут как апплет.

2.3 Обход задач

Какими должны быть исходные URL-адреса, то есть источники, установленные для поискового робота?

На рисунке 2 показана обобщенная задача С. Источники(see) страницы выбираются таким образом, чтобы от каждого источника существовал по крайней мере один путь к целевой странице, на расстоянии не более чем Dзвеньев. На практике мы идентифицируем источник страниц, сначала принимая во внимание inlinks каждого источника страницы. Отправка ссылки: запрос к поисковой двигатель(http://google.yahoo.com ) дает до 20 URL-адресов страниц, ссылки на страницы. Затем повторяем эту процедуру для каждого из этих первыхсоседей( наверное рядом стоящих), и так далее для шагов D. Следовательно, если бы мы имели 10 целей, мы бы в конечном итоге получили10 × 20D уникальных URL-адресов. Подмножество 10 источников URL- выбрали случайным образом из этого множества. Обратите внимание, что эта процедура не гарантирует путь от источника, установленного для каждой цели.

Такая обобщенная задача С формализует класс задач, чьи трудности интуитивно возрастают с увеличением D. При D = 0, источники совпадают с целями так, что поисковые роботы начинают со страниц, которые считаются актуальными. Эта задача имитирует режим поиска"запрос по образцу", где пользователь предоставляет сканеру несколько образцов соответствующих страниц. В качестве альтернативной стратегии эти источники могут быть также получены из веб-поисковой системы. Идея заключается в том, чтобы увидеть, если С способны найти другие релевантные страницы по данной теме. Именно этот подход используется в большинстве оценочных исследований на С до настоящего времени.

Предположение подразумевается в С задаче D = 0, что страницы, как правило, соседствуют друг с другом. Таким образом, цель Сдолжна оставаться сосредоточенной, чтобы оставаться в пределах диапазона, в котором были определены соответствующие документы. Тем не менее, источники, очевидно, не могут быть использованы для оценки рабочих характеристик crawler.Типичные области применения С задачи D = 0 - запрос времени агентов поиска, которые используют результаты поисковой системы в качестве отправной точки, чтобы обеспечить пользователю недавние и персонализированные результаты.

При D> 0, источникиС отличаются от целевых страниц. Так что в данном случае имеется мало предварительной информации, доступной для поисковых роботов о соответствующих страницах, когда начинается сканирование. Идея заключается в том, чтобы увидеть, если С способны найти цели, и / или другие соответствующие страницы. D> 0 включает в себя поиск нужных страниц при запуске С из ссылок.

За некоторыми исключениями, задачи D> 0 редко рассматриваются в литературе, хотя они позволяют моделировать проблемы различной степени сложности. D> 0 задача реальна, так как довольно часто пользователи не могут указать известные соответствующие URL.

Мы считаем, что это важно понимать С в рамках контекста задачи соответственных более чем одно значение D, то есть, с различной степенью сложности, так как они служат разным режимам поиска. Для целей данной статьи мы ограничиваем наш анализ на одну сложную задачу, D = 3. Заметим, что, учитывая большое среднее разветвление веб-страниц, эта проблема немного напоминает поиск иголки в стоге сена. Чтобы проиллюстрировать это, заметим, что в ширину С будет иметь шанс в l^(-D), чтобы посетить цель из источника при среднем разветвленииL звеньев. Начиная с 10 источников и посещение 1000 страниц, l≈ 10 и D = 3 будет означать, ожидаемый верхней границы около 10% об отзыве целевыхстраниц.

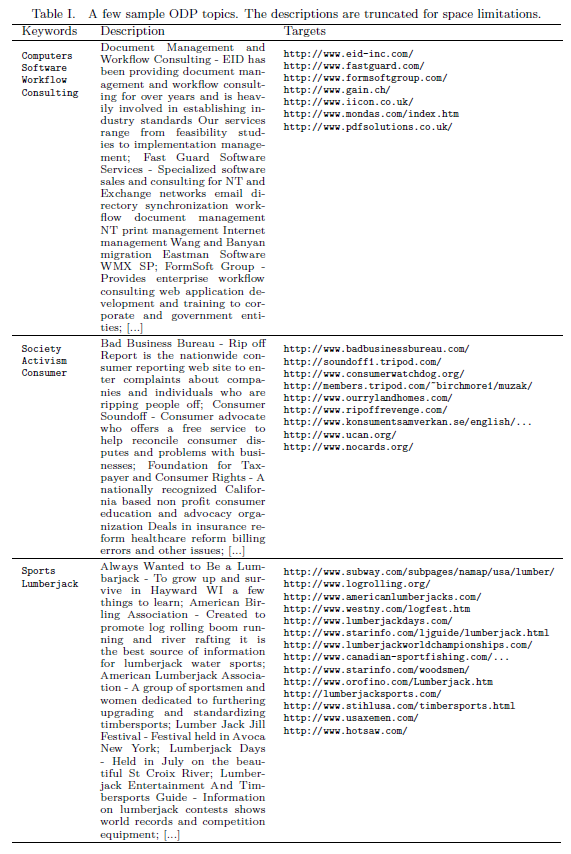
2.4 Темы и цели Dataset

Для оценки алгоритмов С, нам нужны темы и некоторые соответствующие цели. Можно было бы теоретически генерировать темы и целевые наборы, используя частые запросы от поисковых систем и оценок пользователей. Однако такой подход не удобен, учитывая характер динамики Сети. К счастью, эти данные уже доступны из веб-каталогов. В в нашем прошлом исследовании мы использовали Yahoo! из-за своей популярности, в нашей текущей работе мы используем Open Directory Project (ODP), поскольку она имеет меньше коммерческой предвзятости, это «открытый ресурс,"и она поддерживается очень большой и разнообразный ряд редакторов-добровольцев.

Мы собрали темы, запустив рандомизированное в ширину С, начиная с каждой из основных категорий на сайте Open Directory. Списки с пятью или более внешних ссылок используются для получения темы. Тема, представлена ​​тремя типами информации, полученной из соответствующего списка страницы. Во-первых, слова в иерархии ODP формируют ключевые слова этой темы. Во-вторых, внешние ссылки формируют цели этой темы. В-третьих, мы объединяем текстовые описания и якорный текст целевых URL-адресов, чтобы сформировать описание для темы. Разница между ключевыми словами темы и ее описанием, что мы даем в качестве моделей (коротких) запросов, например, по тематике; мы используем последний, который является гораздо более детально представленным о теме, чтобы оценить актуальность просканированных страниц в нашем ретроспективном анализе. В таблице приведены некоторые примеры темы. Эксперименты, описанные в данной работе используют 50 таких тем.(таблица 1).

2.5 Показатели эффективности

В предыдущих исследованиях мы исследовали несколько альтернативных методик оценки С [Menczer 2003; Menczer и др. 2001]. К ним относятся методы оценки значимости страницы с помощью линейных классификаторов, а также с использованием подобия вычислений. Мы также исследовали различные методы для обобщения характеристик С, таких как построение среднего сходства найденных страниц в описаниях тем с течением времени и вычисления средней доли извлекаемых страниц, которые были оценены как «релевантные» по классификаторам. Исходя из нашего предыдущего опыта, в другой статье мы подробно обсудим ряд мер, которые мы выбрали в качестве минимального набора необходимых для обеспечения оценки эффективности работы С [Srinivasan и др. 2002]. В частности, мы предлагаем оценить с помощью целее-



вых и лексические критерии релевантности страницы. Это приводит к набору 4 · D · G , где D является параметром задачи трудности и G является темой общности параметра. Каждая метрика производительности может быть представлена как функция обхода времени, чтобы получить динамическую перспективу. Кроме того мы оцениваем эффективность С алгоритмов - аспект, который ранее не рассматривался в литературе.

В данной работе мы используем только два из показателей эффективности предложенных в Srini- Васан и др. [2002], сосредоточив внимание на влияние различных методов машинного обучения на производительность обхода алгоритмов с течением времени. Эта точка зрения особенно уместна при изучении таких вопросов, как адаптация, последствия которой могут варьироваться в процессе обхода контента. Динамические меры обеспечивают временную характеристику стратегии обхода контента, рассматривая страницы неправдоподобными в то время как С находится в стадии разработки.

Первая метрика простого уровня, основанный на целевых страницах(формула 1)

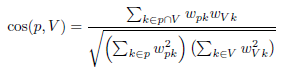


где S (I) представляет собой набор страниц, просмотренных с помощью С, I и T является установленной целью.

Эта мера позволяет определить, насколько хорошо сканер может найти несколько наиболее релевантных страниц. Тем не менее, вполне может быть много других соответствующих страниц, которые не указаны в целевом наборе. Для оценки возможности С, чтобы найти эти другие соответствующие страницы, мы должны оценить значимость любого С страницы. Мы делаем это, используя лексическое сходство между страницей и описанием темы. Напомним, что поисковые роботы не имеют каких-либо знаний в описании темы, и что эти описания вручную составлены экспертами. Поэтому страницы, похожие на эти описания имеют хорошую вероятность быть тесно связанными с темой. Вместо того, чтобы вызывать бинарную оценку актуальности, угадав произвольное пороговое значение подобия, во второй метрики мы измеряем среднее сходство между описанием темы и набором страниц сканироваться:( формула 2),



где V представляет вектор, представляющий описание темы и формула 3

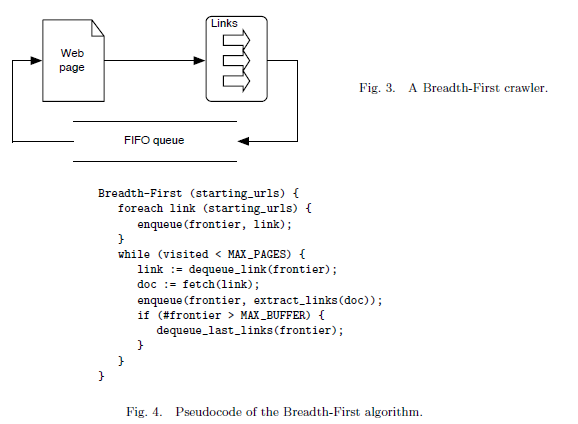


(является стандартной функцией косинус подобия). Здесь WDK является весом TF-IDF терминаk и d, где обратная частота документа вычисляется из множества Ci S(i) когда-то посещенных страниц MAX PAGES каждого С. Оба ρ и среднее сходство σ может быть в виде зависимости от количества страниц С, чтобы получить траекторию в течение долгого времени, которая отображает динамическое поведение С.

3. С АЛГОРИТМЫ

С используем гиперссылкой, чтобы получить новые страницы путем обхода ссылок, ранее извлекаемых из них. Как страницы извлекаются, их внешние ссылки могут быть добавлены в список непосещенных страниц, который упоминается как С границы. Одной из основных задач в ходе прогресса тематического обхода заключается в определении следующей наиболее соответствующей ссылки, чтобы следовать от границы.

Алгоритм выбора следующей ссылки для обхода обязательно привязан к целям С. Индексатор, который направлен на индекс веб будет производить различные виды решений.Первый порядок обхода контента - важен, второй может рассмотреть синтаксис URL, чтобы ограничить восстановленных страниц в домене .edu, а третий может использовать сходство с некоторыми исходными страницами, чтобы направлять выбор канала связи. Даже поисковые роботы, служащие той же цели могут принимать различные стратегии обхода контента.



3.1 С Описание

В этом разделе мы опишем и оценим пять различных алгоритмов, которые мы реализовали в рамках нашей системы оценки:Breadth-First, Best-First, PageRank, Shark-Search, and InfoSpiders. Наш выбор предназначен для широкого представления алгоритмов С, описанных в литературе( с уклоном в сторону тех, которые мы смогли переописать). Один из этих алгоритмов, InfoSpiders, проистекает из нашего собственного предварительного и текущего исследования . В разделе 4 мы обобщим два из этих алгоритмов (Best-First и Shark-Search) через параметризацию их уровень ненасытимости.

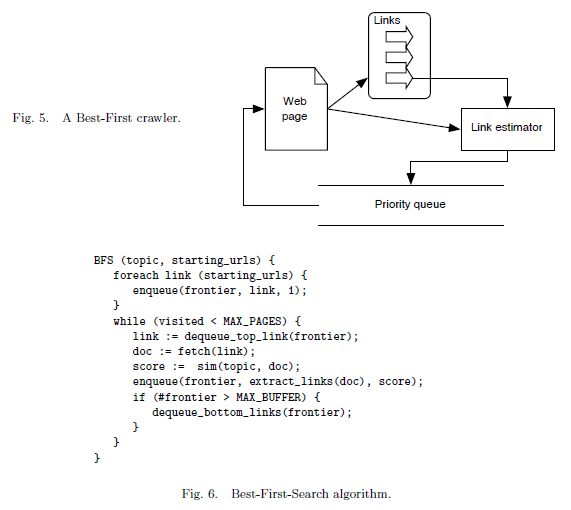
3.1.1 Breadth-First

Breadth-First является простейшей стратегией С. Этот алгоритм был исследован еще в 1994 году в Webcrawler, а также в более поздних исследованиях.

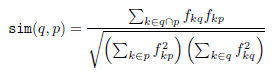
Он использует границу в виде очереди FIFO, С ссылки в том порядке, в котором они встречаются. Обратите внимание, что когда граница заполнена, С может добавить только одну ссылку с С страницы. Breadth-First показан на рисунке 3, алгоритм показан на рисунке 4. Breadth-First используется здесь в качестве базового С; так как он не использует каких-либо знаний о теме, мы ожидаем, что его производительностьобеспечит нижнюю границу для любого из более сложных алгоритмов.

3.1.2 Best-First

Best-First была изучена Cho и др. [1998] и Hersovici и др. [1998]. Основная идея заключается в том, что дается граница ссылок, лучшая связь в соответствии с некоторым критерием оценки выбирается для сканирования. Различные Best-First

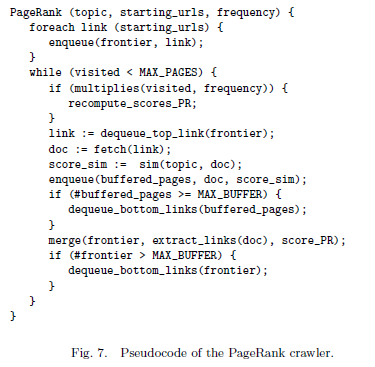


стратегии возрастающей сложности и (потенциально) эффективности могут быть разработаны на базах более сложных критериев оценки ссылок. В нашей реализации, процесс выбора линии связи руководствуется вычислением лексических сходств между ключевыми словами этой темы и страницами источника для ссылки. Таким образом, сходство между страницей и тема ключевых слов используется для оценки значимости страницы, заостренной формы, р. URL с наилучшей оценкой выбирается для сканирования. Косинусное подобие используется обходчиком и ссылки с минимальным счетом подобия удаляются от границы, если это необходимо, чтобы не превысить предельный размер MAX BUFFER. На рисунке 5 показана первая Best-First, а на рисунке 6 предлагается упрощенный псевдокод алгоритма Best-First-Search (BFS). Функция SIM



возвращает косинус сходство между темой и страницей:(формула 4), где q-тема, р -извлеченная страница

FKD -частота цели к в д.



Предварительные эксперименты при D = 0 задачи показали, что BFS является конкурентным С [Menczer и др. 2001], поэтому здесь мы хотим оценить, как он сравнивает с другими алгоритмами в более сложной задаче обхода.

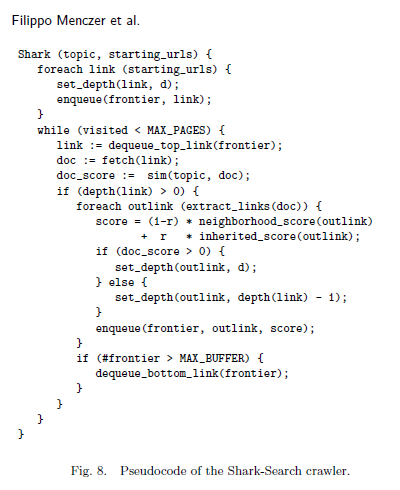
3.1.3 PageRank.

PageRank был предложен Брин и Пейдж [1998] в качестве возможной модели поведения пользователей. PageRank представляет собой вероятность того, что случайный человек(тот, кто следует ссылки случайным образом из одной страницы на другую) будет на этой странице в любой момент времени. Счет на страницу зависит рекурсивно от десятков страниц, которые указывают на него. (Формула 5),



где р -страница, которую набрал в множество страниц, указывающих на р, представляет собой набор связей из D, а константа γ <1 - коэффициент затухания, который представляет собой вероятность того, что случайный человекпопросит другую случайную страницу. Первоначально предложенный PageRank был предназначен для использования в сочетании с содержанием на основе критериев для ранжирования извлекаемых наборов документов . PageRank используется в поисковой системе Google. Совсем недавно PageRank использовался для руководства С [Cho и соавт. 1998] и для оценки качества страницы [Henzinger и др. 1999].

В предыдущей работе мы оценивали С, основанный на PageRank [Menczer и др. 2001]. Мы использовали эффективный алгоритм для вычисления PageRank [Haveliwala 1999]. Тем не менее, как видно из приложения 5, PageRank требует рекурсивного вычисления до сходимости, и, следовательно, его вычисление может быть очень ресурсоемким процессом. В идеальной ситуации следует пересчитывать PageRanks каждый раз, когда URL должен быть выбран из границы. Вместо того, чтобы повысить эффективность, мы пересчитываем вес PageRank через регулярные промежутки времени.

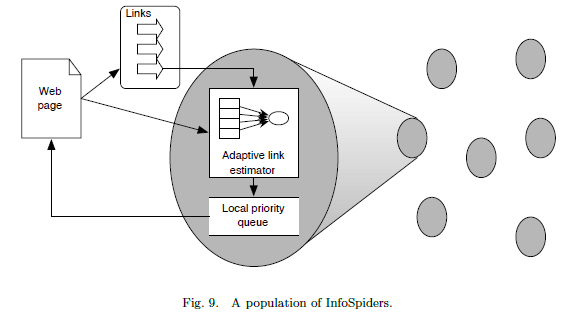


PageRank на С можно рассматривать как вариант Best-firstс другой функцией оценки линии связи (рисунок 5). Алгоритм иллюстрируется на рисунке 7. SIM () функция возвращает косинус сходство между ключевыми словами для темы и страницы, как измерено в соответствии с уравнением 4, и PageRank вычисляется в соответствии с уравнением 5. Обратите внимание, что алгоритм PageRank должен поддерживать структуру данных посещаемых страниц в дополнение к границе, чтобы вычислить вес PageRank. Этот буфер может содержаться в большинстве страниц MAX Buffer. Мы предполагаем,страницы с 0 степенью (например, URL-адреса в границе), чтобы были неявно связаны с каждой страницей в буфере, как это требуется для алгоритма PageRank. Мы используем γ = 0,8 и порог сходимости устанавливается в размере 0,01.

3.1.4 Shark-Search.

Это более агрессивная версия Fish-Search . В Fish-Search С поиск более широк в областях Сети, в которых были найдены соответствующие страницы. В то же время, алгоритм поиска прекращается в тех регионах, которые не выдают соответствующие страницы. Он предлагает два основных улучшения по сравнению с Fish-Search. Он использует непрерывную функцию для оцененных измерений релевантности в отличие от двоичной функции релевантности в Fish-Search. Кроме того, Shark-Search имеет более утонченное понятие потенциальных баллов для ссылок в обход границы. Потенциальная оценка ссылок влияет на якорный текст, текст окружающий ссылки.

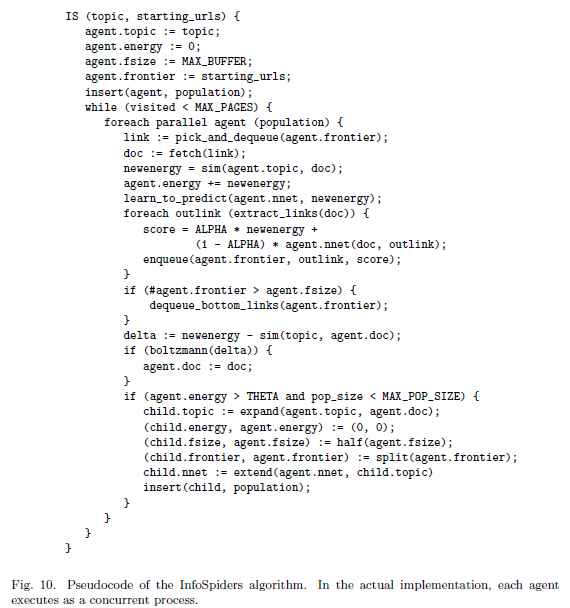
Shark-Search можно рассматривать как вариант Best-first с более сложной функции оценки линии связи (рисунок 5).



На рисунке 8 показан псевдокод алгоритма Shark-Search. Параметры D и R означают, соответственно, максимальную глубину и относительную важность( наследуется по сравнению с соседских баллов). В нашей реализации мы устанавливаем D = 3 и г = 0,1.

3.1.5 InfoSpiders.

В InfoSpiders используется адаптивная популяция агентов поиска страниц, имеющих отношение к данной теме с использованием эволюционирующих векторов запросов и нейронных сетей, чтобы решить, каким ссылкам следовать. Идея иллюстрируется на рисунке 9. Этот эволюционный подход использует меру соответствия, основанную на сходстве в качестве локального критерия выбора. Оригинальный алгоритм (смотри, например, Menczer и Бэлью [2000]) был ранее упрощен и реализован как модуль С [Menczer и др. 2001]. Этот С отсутствовал в исполнении BFS из-за его локального поведения и отсутствия какой-либо памяти. С тех пор был сделан ряд улучшений к оригинальному алгоритму, сохраняя при этом способность узнать оценки линии связи с помощью нейронных сетей и сосредоточить поиск в сторону более перспективных направлений путем селективного размножения. Полученный новый алгоритм схематично описан в псевдокоде на рис.10.



InfoSpiders агенты являются независимыми друг от друга и растут параллельно. Адаптивное представление каждого агента состоит из списка ключевых слов (инициализируется с темой ключевых слов) и нейронной сети, используемой для оценки новых ссылок. Каждый входной блок нейронной сети получает счетчик частоты, с которой ключевое слово происходит в непосредственной близости от каждого звена( должна перемещаться, чтобы придать большее значение для ключевых слов), (происходящих вблизи линии). Существует единый модуль вывода. Одним из упрощений, которые мы сохранили в этой реализации является то, что нейронные сети не имеют скрытых слоев, поэтому состоят из простых персептронов. Это означает, что, хотя выходной блок вычисляет нейронную сеть по нелинейной сигмоидальной функции входных частот, нейронная сеть может только узнать линейные разделения входного пространства. Тем не менее мы считаем, что это достаточно мощное представительство усваивания уровней корреляции между релевантностью и распределением ключевых слов вокруг ссылок. Выход нейронной сети используют в качестве оценки численного качества для каждой линии связи, рассматриваемой в качестве входных данных. Эти оценки затем в сочетании с оценками, основанными на косинус подобия (уравнение 4) между вектором ключевых слов агента и страницу, содержащую ссылки. Параметр α, 0 ≤ α ≤ 1 регулирует относительную важность, которая придается оценкам на основе нейронной сети по сравнению с родительской страницей. На основе комбинированной оценки, агент использует стохастический селектор, чтобы выбрать одну из ссылок с вероятностью( формула 6),



где λ является ссылкой со страницы р и μ (Х) является его комбинированным счетом. Параметр βрегулирует алчность(?) селектора канала. В экспериментах, описанных в данной работе мы используем α = 0,5 и р = 5.

После того, как новая страница была получена/выбрана, агент получает энергию пропорционально сходству между его ключевыми словами вектора и новой страницы (уравнение 4). Нейронная сеть агента может быть обучена, чтобы улучшить оценки ссылок путем прогнозирования сходства новой страницы, учитывая входы со страницы, содержащего ссылку, ведущую к ней. Мы используем эпоху(?) стандартного обратного распространения ошибки со скоростью обучения 0,5. Такая методика обучения обеспечивает InfoSpiders с уникальной способностью адаптировать ссылку в процессе обхода контента.

Агент переходит к вновь выбранной страницы, только если функция Больцмана возвращает истинное состояние (см Рисунок 10). Это определяется стохастически на основе вероятности( формула 7),



где δ -разница между сходством новой и текущей страницы вектора ключевых слов агента; Т = 0,1 и является показателем температуры.

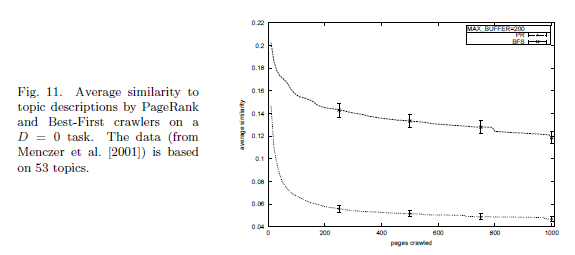
Используется для определения того, должен ли агент воспроизвести после посещения страницы. Агент воспроизводит, когда уровень энергии проходит постоянный порог THETA = 2. При воспроизведении, потомство получает половину линии границы родителя, вектор ключевых слов потомства также мутировали, добавив термин, который наиболее часто встречается в текущем документе родителя. Такая мутация обеспечивает InfoSpiders уникальную способность к адаптации стратегии поиска на основе новых ключей, захваченных из перспективных страниц, в то время как воспроизведение само по себе позволяет населению предвзятости поиска к районам (агентов), которые приводят к хорошим страницам.

MAX POP SIZE = 8 агентов( поддержание распределенной границы которого общий размер не превышает MAX Buffer). Поведение отдельных агентов по-прежнему базируется на локальных оценках ссылок и ключевых слов, однако в отличие от оригинального алгоритма, каждый агент сохраняет более глобальное представление пространства поиска .

Другие использования эволюционных алгоритмов для задач веб-поиска. Например, генетический алгоритм был применен к эволюции популяций ключевых слов и логических операторов, используемых для сбора и рекомендовать страницы пользователю [Moukas и Захария 1997; Ник и Фемида 2001]. В таких системах процесс обучения поддерживается мета-поисковой системой и руководствуется обратной релевантности от пользователя, в то время как в этой статье мы заинтересованы в средствах, которые действительно могут сканировать веб и учиться неконтролируемо взаимодействовать с окружающей средой.

3.2 Экспериментальные результаты

Приведем теперь результаты ряда обходов, проведенных для оценки и сравнения производительности, эффективности и масштабируемости алгоритмов С, описанных в разделе 3.1.

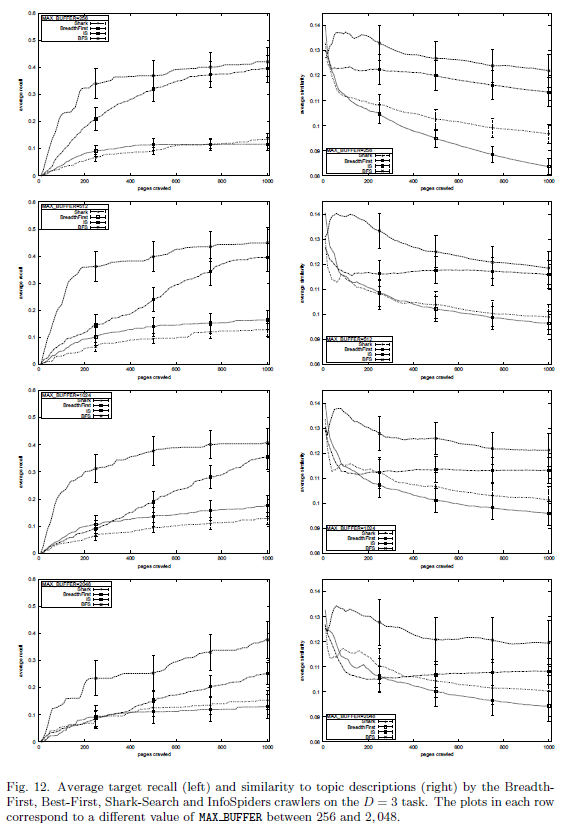


Если не указано иное, результаты в этом и следующих разделах основаны на D = 3 С задачи и параметр MAX СТР установлен в 1000. Эти параметры соответствуют особенно сложной задачи, в которой соответствующие страницы находятся далеко, и человек может посетить только очень небольшое количество страниц - возможно, в то время как пользователь ожидает. Как уже говорилось в разделе 2.3 эта сложная задача сродни поиску иголки в стоге сена, что делает ее интересной тестовой задачей.

Кроме того, если не указано иное, результаты в этом и следующих разделах основаны на средних показателях по 50 темам. На протяжении всей работы, столбики ошибок на графиках соответствуют ± 1 стандартное отклонение, т.е. стандартное отклонение от среднего значения по всей тематике. Мы интерпретируем неперекрывающиеся планки погрешностей в качестве статистически значимой разницы в уровне достоверности 68%, предполагая нормальную помеху.

3.2.1 Динамические характеристики.

В наших предыдущих экспериментах было обнаружено, что PageRank поисковик не конкурентоспособен, в связи с его минимальной эксплуатацией контекста темы иограниченностью ресурсов. Метрика PageRank разработана в качестве глобальной меры и ее ценности имеют мало пользы, когда вычисления происходят над небольшим набором страниц. Для иллюстрации, рисунок 11



среднее сходство с описаниями в разделе D = 0 , т.е., начиная с целевых страниц. Учитывая низкую эффективность PageRank, производительность и задачи, которые мы используем здесь, мы не будем в дальнейшем работать по этому алгоритму.

Выполнение остальных четырех С - Breadth-First, Best-First, Shark- Search and InfoSpiders- сравнивается на рисунке 12. Мы провели четыре варианта каждого алгоритма обхода для значений MAX BUFFER между 28 = 256 и 211 = 2, 048,6.

Как и следовало ожидать Breadth-Firstработает плохо, и, таким образом, представляет собой основу. Удивительно, но Shark- Search выполняет не значительно лучше, чем Breadth-First за исключением одного случая, метрики подобия для MAX BUFFER = 256. Shark-Search -этот алгоритм не может быть достаточно исследован (смотри раздел 4). Best-First и InfoSpiders значительно превосходят другие два в конце сканирований, для подобия метрики в MAX BUFFER = 2048 случае, когда InfoSpiders страдает большим первоначальным убытком, из которого она не полностью восстановилась после 1000 страниц.

В то время как Best-First показывает преимущество над InfoSpiders в ранней стадии обходов. На начальных этапах BFS помогает его «жадность», в то время как InfoSpiders платит за адаптируемось - нейронные сети не обучены, эволюционный уклон еще не имел шанса умереть, и стохастический селектор делает некоторые локально неоптимальные варианты. Однако эти возможности оказываются более полезными на более поздних этапах ползания. Например, сходство с описаниями темы слегка улучшается для InfoSpiders в 3-х из 4-х графиках, а все остальные С показывают тенденцию к снижению после 200 страниц.

3.2.2 Масштабируемость.

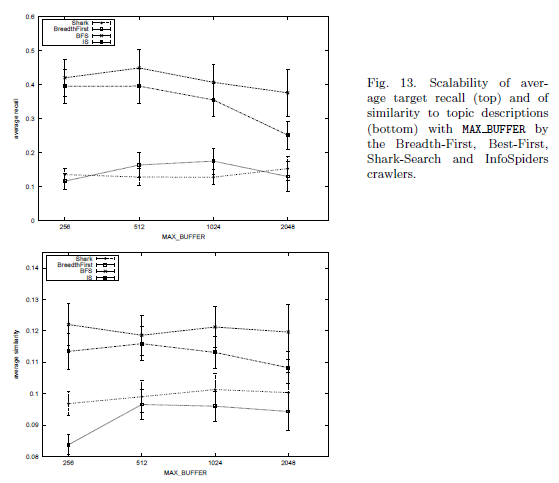
На рисунке 12 видно, что производительность зависит от значения параметра MAX BUFFER. Обратите внимание, что даже если MAX BUFFER > MAX PAGES, производительность может зависеть от пограничного размера, так как количество ссылок, с которыми сталкиваются поискавики(С), как правило, намного больше, чем количество посещенных страниц. Но в то время как ограниченный размер буфера вынуждает любойпоисковик удалить множество ссылок от его границы, которые устранен ссылки зависит от алгоритма поисковика. Для того, чтобы лучше оценить различные алгоритмы масштабирования, С с пространством (памяти) ресурсов, доступных для них, производительность (Рисунок 13) участков по сравнению MAX Buffer.

Breadth-First, Shark-Search and Best-First не будут затронуты пограничным размером, но с одним исключением: Сходство на тему описаний значительно увеличиваетBreadth-First , когда MAX BUFFER = от 256 до 512. Это свидетельствует о том, что ограничение пространственных средств повредит алгоритму больше, если он не может расставить приоритеты на свои связи, в связи с повышенным риском выбрасывает хорошие ссылки. Для InfoSpiders, производительность уменьшается с увеличением размера границы. Мы обсудим этот эффект в разделе 7.

3.2.3 Эффективность.

Разберем временную сложность различных С алгоритмов. Мы делаем ставку на процессорнное время, так как все сканеры одинаково подвержены сети ввода / вывода задержки. Мы также исключаем процессорноевременя, принятых общих функций полезности, разделяемых всеми С (см Раздел 2.1).

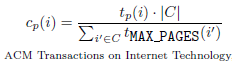
Мы можем оценить эффективность только наших собственных реализаций сканерами. Рассмотрим, например, структуру данных, используемую для реализации очереди пограничного приоритета в Best-First алгоритме. Основные операции вставки и удаления ссылок из очереди наиболее эффективно поддерживается множеством O(log(MAX BUFFER)) . Однако, если недавно извлеченный URL уже находится в границе, это создаёт линейную сложность в множестве, и должно быть сделано для каждой ссылки в недавно сканированной странице. Alternatively, using an associative array or hash table, this checking operation is cheap (O(1)) but inserting and removing links requires an expensive sorting operation (O(MAX BUFFER · log(MAX BUFFER))) once per page crawled. The asymptotic complexity of priority queue operations per crawled page is O(L · MAX BUFFER)

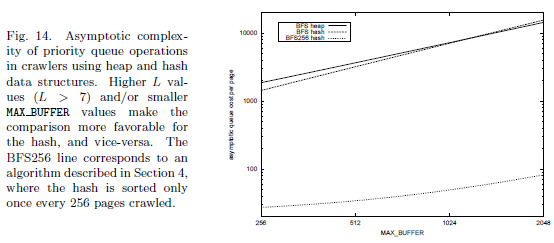


for the heap and O(L+MAX BUFFER·log(MAX BUFFER)) for the hash, where L is the average number of links per page. Figure 14 plots the actual costs for L = 7, which is an accurate estimate for the Web [Kumar et al. 2000].

График показывает, что в то время как в диапазоне MAX BUFFER, используемом в наших экспериментах две структуры данных дают очень похожую эффективность, хэш является предпочтительным для усовершенствованных поисковиков, описанных в разделе 4. По этой причине мы используем ассоциативный массив для хранения ссылок границы, не только в Best-First, но и в других поисковых роботах. Тем не менее, дальнейшая оптимизация может быть возможной для некоторых С, следовательно, мы не можем утверждать, что наш анализ эффективности обязательно применим к другим реализациям.

Еще один нюанс в том, что фактическое время, затраченное на любой алгоритм не имеет никакого значения, поскольку оно во многом зависит от конфигурации нагрузки и аппаратной части машины, используемых для запуска обходов - даже эксперименты, описанные в данной работе, были выполнены на большом количестве машин с самыми различными аппаратными конфигурациями и меняющимися нагрузками. Для того, чтобы получить меру сложности времени, которое может быть конкретно сравнивать эффективность С, мы вводим относительную стоимость Сi как соотношение между временем, принятым I для посещения р страниц и среднее суммарное время, затраченное набором поисковым роботjм посетить MAX Количество страниц:(формула 8),





где С представляет собой набор С и i∈C.

Относительная стоимость метрики является надежной волной относительно различных тем из разных машин, до тех пор, как все поисковики запустят одну тему из той же самой машины.

Еще одна полезная метриката, которая сочетает в себе два источника доказательств для исполнения с эффективностью. С этой целью определим производительность / стоимость метрики P как соотношение между производительностью и относительной стоимости I, где предыдущий дается произведением среднего целевого вызова и среднего сходства с описаниями темы после р страниц:( формула 9),



где Q - множество тематических разделов.

Рисунок 15 демонстрирует С1000 и P1000 MAX BUFFER для четырех поисковых роботов.

Shark-Search и является самым дорогим С. Breadth-First является наиболее эффективным. Breadth-First становится более эффективным для увеличения размера приграничной, поскольку она должна урезать свою границу. Best-Firstимеет промежуточную сложность. Это становится менее эффективным с увеличением пограничного размера, потому что есть больше ссылок для сравнения. Что самое удивительное, InfoSpiders почти так же эффективен, как Breadth-First, с двумя временными сложностями, сходящихся для MAX BUFFER = 2,048.

Производительность / стоимость соотношения на рисунке 15 делает InfoSpiders победителем благодаря своей производительности и отличной эффективности. Существует также четкая положительная тенденция с увеличением размера границы.

# 3 Заключительный этап

В данной статье рассмотрены разные поисковые машины. Проведены примеры их алгоритмов произведены их сравнения по важным критериям, таким как масштабируемость. Так же проведены оценки времени их выполнения.