УДК 004.738.5:519.7
ББК 32.973-22.18

# АДАПТИВНЫЙ КРАУЛЕР ДЛЯ ПОИСКА И СБОРА ВНЕШНИХ ГИПЕРССЫЛОК

**Печников А. А.**[1],
(*Учреждение Российской академии наук
Институт прикладных математических исследований
КарНЦ РАН, Петрозаводск*)
**Чернобровкин Д. И.**[2]
(*Факультет прикладной математики -
процессов управления Санкт-Петербургского
государственного университета, Санкт-Петербург*)

*Описывается поисковый робот* (*краулер*)*, предназначенный для сбора информации об исходящих гиперссылках с задаваемого множества сайтов, относящихся к одной тематике. Адаптивное поведение краулера сформулировано в терминах задачи о многоруком бандите. Проведенные эксперименты показывают, что выбор адаптивного алгоритма рационального поведения краулера зависит от тематики множества сайтов.*

Ключевые слова: гиперссылка, поисковый робот, адаптивное поведение, задача о многоруком бандите, индексы Гиттинса.

## *1. Введение*

Концептуальная модель фрагмента Веба [2] строится на основе задаваемого множества веб-сайтов, относящихся к одной тематике и являющихся регламентируемыми, т.е. создающихся и развивающихся по заранее сформулированным правилам, утвержденным в виде нормативных документов организаций-

[1] *Андрей Анатольевич Печников*, *доктор технических наук*, *доцент* (*pechnikov@krc.karelia.ru*).
[2] *Денис Игоревич Чернобровкин*, *аспирант* (*denis_univer@mail.com*).

владельцев ресурсов. Примерами тематических целевых множеств служат официальные веб-сайты институтов Санкт-Петербургского научного центра РАН и официальные веб-сайты высших учебных заведений Санкт-Петербурга. Анализ внешних гиперссылок, сделанных с веб-сайтов целевого множества, позволяет построить так называемое «сопутствующее множество» – множество сайтов, не входящих в целевое множество, на которые имеются гиперссылки с целевого множества. Сопутствующее множество разбивается на несколько непересекающихся подмножеств, в зависимости от степени связности входящих в них сайтов с сайтами целевого множества. Концептуальная модель фрагмента Веба представляет собой набор множеств, состоящий из целевого множества и подмножеств сопутствующего множества, заданных на них отношениях, определяющих структуру фрагмента Веба, а также атрибутов множеств и отношений, задающих их характеристики (термин «концептуальная модель» понимается в смысле работы [5]). Реализация концептуальных моделей для различных фрагментов российского Веба [1, 3, 4] позволяет сделать выводы об их организации и предложить решение ряда задач по улучшению их присутствия в Вебе.

В более широком контексте построение и исследование концептуальных моделей фрагментов Веба относится к такому направлению вебометрики, как гиперссылочный анализ (*link analysis*) [12], существенное место в котором отводится краулерам (поисковым роботам), – специализированным программам, использующим графовую структуру Веба для перемещения по страницам веб-сайтов с целью сбора требуемой информации. В общем случае задача краулинга может рассматриваться как многоцелевая задача поиска с ограничениями, где большое разнообразие целевых функций вместе с нехваткой соответствующих знаний о месте поиска делает задачу весьма сложной [9].

Возвращаясь к теме построения концептуальной модели фрагмента Веба для заданного тематического целевого множества, заметим, что её основой является база данных внешних гиперссылок, связывающих веб-сайты модели. Краулер, формирующий в процессе сканирования веб-сайтов такую базу дан-

ных, очевидно, должен быть избирательным и адаптивным [9]. Избирательность краулера в нашем случае определяется его направленностью на поиск внешних гиперссылок (с некоторой дополнительной информацией о них, например, контекстом гиперссылки). Адаптивность краулера заключается в том, что за заданное (ограниченное) время должно быть найдено максимально возможное количество внешних гиперссылок, что обеспечивает более полный охват веб-сайтов, входящих в сопутствующее множество.

В статье описана структура и базовые функции разработанного краулера *BeeBot* («робот-пчела»), предназначенного для сбора внешних гиперссылок с веб-сайтов заданного целевого множества. Задача о нахождении максимального количества внешних гиперссылок за конечный период времени сформулирована как задача о многоруком бандите [6] и предложены три алгоритма её решения. Были проведены сравнительные эксперименты на множестве официальных веб-сайтов университетов и научных институтов Санкт-Петербурга, показывающие, что выбор алгоритма рационального поведения краулера во многом зависит от тематики сканируемого множества сайтов. Этот результат затем используется для управления процессом сканирования *BeeBot*: для официальных веб-сайтов университетов и для веб-сайтов научных институтов используются различные алгоритмы адаптивного поведения.

## 2. BeeBot – краулер для сбора внешних гиперссылок

Работу краулера можно описать следующим образом [9]: сканирование сайта начинается с начальной страницы и затем робот использует ссылки, размещенные на ней, для перехода на другие страницы. Каждая страница сайта анализируется на наличие требуемой информации, которая копируется в соответствующее хранилище в случае обнаружения. Процесс повторяется до тех пор, пока не будет проанализировано требуемое число страниц либо пока не будет достигнута некая цель. Такое описание работы краулера является слишком общим, и далее мы

конкретизируем его для специфических моментов, связанных с разработкой и реализацией *BeeBot*.

Далее нам понадобится следующее определение уровня страницы веб-сайта: считаем, что начальная страница сайта имеет уровень 0, а уровень любой другой страницы сайта – это минимальное количество внутренних гиперссылок, ведущих от начальной страницы к данной.

*BeeBot* состоит из трех основных частей: база данных, блок управления краулером (включая интерфейс пользователя и представление результатов) и блок сканирования и обработки данных. В качестве основы для базы данных была выбрана бесплатная версия СУБД *Microsoft SQL Server* 2008 R2 *Express Edition*, поскольку с ней легко интегрируются приложения платформы *.NET*. В частности, используется технология *Entity Framework*, позволяющая представлять данные в виде объектов, что упрощает работу с БД.

Для хранения информации используется 5 таблиц: *Crawler Settings*, *Seeds*, *External Link*, *Internal Links*, *Bad Links*. В таблице *Crawler Settings* хранятся настройки для краулера. Например, значение одной из настроек ограничивает максимальный уровень страниц, до которого может дойти краулер в процессе сканировании веб-сайта. Это значение устанавливается по умолчанию и распространяется на все сканируемые сайты.

Таблица *Seeds* содержит начальную информацию обо всех сканируемых веб-сайтах. Начальная информация о каждом веб-сайте задается пользователем и представляет собой номер сайта, полное название сайта, его краткое название и начальный адрес – доменное имя сайта, с которого начинается сканирование. В таблице *Internal Links* хранятся сведения о внутренних ссылках, найденных на сканируемых сайтах. Они описываются собственно самой ссылкой, начальным адресом, страницей, на которой найдена внутренняя ссылка, уровнем этой страницы и её статусом, показывающим, обработана ли страница, соответствующая внутренней ссылке, краулером или нет. Таблица *External Links* содержит информацию обо всех найденных внешних гиперссылках. В этой таблице поля во многом анало-

гичны таблице *Internal Links*, за исключением того, что у внешней ссылки нет статуса обработки. Кроме того, добавлено поле, содержащее контекст гиперссылки (в частном случае – её анкор). Таблица *Bad Links* содержит так называемые «плохие ссылки». Например, это ссылка, которую краулер не смог воспринять, как корректную ссылку, хотя этот элемент был под тегом *href*. Также сюда могут попасть фрагменты *javascript*-кода, так как иногда веб-мастера вставляют его прямо в тело ссылки.

Рассмотрим блок управления *BeeBot* и пользовательский интерфейс. Пользователю предоставляются следующие возможности по управлению краулером: пуск/остановка работы краулера (надо отметить, что в случае прерывания краулер продолжит с места остановки, не потеряв ссылок), просмотр текущего списка сайтов, добавление новых и удаление ненужных сайтов. Также пользователю дается возможность просматривать собранные ссылки. Для этого разработано специальное окно, содержащее две отдельные страницы, где можно просматривать все собранные внешние и внутренние ссылки, фильтровать результаты и удалять выбранные ссылки. Просмотр результатов проходит в режиме «реального времени», т.е. прямо во время работы краулера: при нажатии на кнопку «*Refresh*» результаты будут обновлены.

Для управления последовательностью сканирования сайтов в состав блока управления входит модуль-планировщик, который в простейшем варианте работает следующим образом: после завершения сканирования очередного сайта из текущего списка выбирается следующий сайт. (Естественное завершение сканирования каждого сайта в этом случае возможно по двум причинам: сайт отсканирован полностью или отсканированы все его страницы до установленного максимального уровня).

Перейдем к блоку сканирования. Процесс сбора ссылок происходит следующим способом: из базы данных берутся начальные адреса всех сайтов и для каждого из них последовательно запускается процесс сканирования по обработке опреде-

ленного количества страниц с целью нахождения внешних гиперссылок.

Обработка страницы начинается с того, что краулер предпринимает попытку загрузки страницы и преобразует её в *XML*-документ с помощью библиотеки *HTMLAgilityPack*. Если в процессе стандартной процедуры загрузки произошел сбой или автоматически была выбрана неправильная кодировка, то страница загружается альтернативным методом, а именно, скачивается в бинарном формате и затем преобразуется в текстовый формат с уже определенной краулером кодировкой. Альтернативная загрузка нужна и в том случае, когда веб-сервер поддерживает кусочную (*chunked*) передачу данных [7, п. 3.6.1], поскольку стандартный загрузчик в данном случае скачивает только первый фрагмент данных, в котором может оказаться не весь *XML* -документ и неполная информация о странице.

Если процедура загрузки не выполняется, то сайт помечается в базе данных как *Not Available* и далее не обрабатывается. В случае если процедура загрузки прошла успешно, краулер считывает полученный *XML*-документ и собирает все доступные фреймы со страницы. Адреса фреймов, в свою очередь, также рассматриваются как внутренние ссылки и сохраняются в базе данных для будущей обработки.

После сбора фреймов краулер собирает все возможные ссылки со страницы (элементы с тегом «*href*»). Перед занесением в базу данных ссылка нормализуется. Как показано в [10], нормализация ссылки в виде приведения её как строки текста к нижнему регистру, а также удаления символа «\» в конце ссылки, приводит к потере менее процента уникальных ссылок, но при этом ведет к избавлению от 50% «семантических дубликатов» (ссылок, которые синтаксически не равны друг другу, но указывают на один и тот же ресурс). Такая нормализация используется в *BeeBot*, поскольку она не изменяет функционала гиперссылки.

Далее, в зависимости от того, к какому типу относится ссылка, она помещается в одну из таблиц *Internal Links* или *External Links* (есть и третий случай, когда «плохая» ссылка

помещается в таблицу *Bad Links*). Решение о том, в какую именно таблицу занести ссылку, краулер принимает на основе доменного имени, которое он извлекает из ссылки. Если доменное имя извлеченной ссылки совпадает с доменным именем сканируемого сейчас сайта, значит ссылка внутренняя, если нет, то внешняя. После завершения обработки страницы краулер извлекает из базы данных следующую необработанную внутреннюю ссылку с самым высоким уровнем для данного сайта. Таким образом, сканирование сайта можно представить как обход виртуального дерева сайта по принципу «вначале вширь».

В идеальном случае работа краулера завершается в том случае, когда будут полностью отсканированы все сайты из заданного списка. В реальности работа может быть завершена по истечении выделенного процессорного времени. На практике такой вариант завершения работы имеет недостатки. Например, если в списке сканируемых сайтов первым окажется сайт РАН (www.ras.ru), есть опасность затратить на него всё отведенное время. Естественным развитием модуля-планировщика является процедура управления последовательностью сканирования сайтов, последовательно выделяющая каждому сайту из текущего списка определенный квант времени для его сканирования. В этом случае даже по истечении выделенного процессорного времени будут частично (или полностью) отсканированы все сайты из списка.

## *3. Адаптивные алгоритмы*

Вернемся к задаче, сформулированной во Введении: за ограниченное время необходимо найти максимально возможное количество внешних гиперссылок с веб-сайтов из заданного списка. Мы уже исключили из рассмотрения случай полного сканирования всех сайтов. Понятно также, что поведение краулера типа «на каждом сайте из целевого множества обрабатывается одинаковое количество страниц» вряд ли приводит к решению этой задачи. Количество внешних гиперссылок, находящихся на странице становится известным только после её обработки, поэтому выбор того или иного веб-сайта для скани-

рования его очередных страниц должен осуществляться на основе текущей (накапливаемой) информации. Таким образом, мы приходим к задаче, известной как задача о «многоруком бандите». Сформулируем её применительно к нашему случаю, используя обозначения и терминологию работ [6, 8].

В базовой формулировке задача о многоруком бандите определяется через случайные величины $X(i, j)$ для $1 \le i \le K$ и $j \ge 1$, где каждый индекс $i$ соответствует игровому автомату, $j$ – порядковому номеру игры с этим автоматом. Последовательная игра на автомате $i$ приносит игроку доходы $X(i, 1)$, $X(i, 2)$, ..., которые представляют независимые одинаково распределенные величины с неизвестным законом распределения и неизвестным математическим ожиданием. Независимость справедлива также и для выигрышей на различных машинах, т.е. $X(i, s)$ и $X(l, t)$ являются независимыми величинами (и обычно с различными распределениями) для любых $1 \le i < l \le K$ и $s, t \ge 1$. В нашем случае «игроком» является краулер, а переменные $X(i, 1)$, $X(i, 2)$, ..., $X(i, j)$, …, обозначают количество найденных на $i$-м сайте внешних гиперссылок на 1-й, 2-й, $j$-й и т.д. сканируемой странице (в порядке обхода «вначале вширь», как говорилось ранее), а $K$ соответствует мощности сканируемого целевого множества. В такой постановке мы делаем допущение о том, что время сканирования любой страницы любого сайта одинаково, поэтому вместо суммарных затрат времени, выделяемых на сканирование всех сайтов, можно ввести суммарное количество планируемых для сканирования страниц, которое обозначим $N$. Из содержательной постановки следует, что $N >> K$. Доход на одном шаге в этом случае равен количеству внешних гиперссылок, найденных на странице, поэтому $X(i, j) \ge 0$ для $1 \le i \le K$ и $1 \le j \le N$.

Решающее правило $R$ – это алгоритм, который выбирает следующий сайт для сканирования, основываясь на последовательности предыдущих сканирований и полученного количества гиперссылок.

Пусть $T(i, n)$ означает количество страниц, отсканированных на сайте $i$ за $n$ сканирований на всех сайтах с использова-

нием правила $R$. Тогда суммарное количество внешних гиперссылок, найденных за $n$ сканирований при использовании правила $R$, равно

$$S^R(n) = \sum_{i=1}^{K} \sum_{j=1}^{T(i,n)} X(i,j).$$

Задача заключается в разработке такого правила $R^*$, которое максимизирует значение функции $S^R(n)$ для заданного конечного горизонта сканирования $N$.

Известно, что в общем случае задача о многоруком бандите является $\mathcal{NP}$-полной задачей [12], поэтому для её решения предложены различные подходы с целью получения хороших результатов при относительно небольших затратах ресурсов. Далее мы описываем и сравниваем три решающих правила $R_1$, $R_2$ и $R_3$, первое из которых является тривиальным, второе построено с использованием индексов Гиттинса [8], а третье основано на алгоритме *UCB*1, предложенном в работе [6].

Правило $R_1$ («тривиальное правило»).

1. На каждом из $K$ сайтов целевого множества последовательно сканировать первые $[N/K]$ страниц ([*] – округление до ближайшего целого).

2. Оставшиеся $N - [N/K]$ страниц сканировать на сайте с номером $i^*$, для которого $i^* = \arg\max_{1 \le i \le K} \sum_{j=1}^{[N/K]} X(i,j)$.

3. $S^{R1}(N) := \sum_{i=1}^{K} \sum_{j=1}^{[N/K]} X(i,j) + \sum_{j=[N/K]+1}^{N-[N/K]} X(i^*,j)$.

4. Стоп.

Правило $R_2$ (индекс Гиттинса).

1. Задать шаг сканирования $n$; $n << N$.

Для всех $i$: $s(i) := 0$ ($s(i)$ – промежуточные суммы, вспомогательные переменные).

2. На каждом сайте отсканировать $n$ страниц.

3. Для всех $i$: $s(i) := s(i) + \sum_{j=1}^{n} X(i, j)$.

4. Для всех $i$: $Ind(i) := s(i)/n$; $t(i) := n$ ($Ind(i)$ – текущие значения индексов Гиттинса для каждого сайта; $t(i)$ – количество отсканированных страниц на каждом сайте).

5. Найти $i^* := \arg\max_{1 \le i \le K} Ind(i)$.

6. Отсканировать $n$ страниц на сайте $i^*$.

7. $s(i^*) := s(i^*) + \sum_{j=t(i^*)}^{t(i^*)+n} X(i^*, j)$; $t(i^*) := t(i^*) + n$; $Ind(i^*) := s(i^*)/t(i^*)$.

8. Если $\sum_{i=1}^{K} t(i) < N$, то перейти к п. 5,

9. $S^{R2}(N) := \sum_{i=1}^{K} \sum_{j=1}^{t(i)} X(i, j)$

10. Стоп.

Правило $R_3$ (алгоритм *UCB*1).

1. Задать начальный шаг сканирования $m$; $m << N$.

Для всех $i$: $s(i) := 0$ ($s(i)$ – промежуточные суммы, вспомогательные переменные).

2. На каждом сайте отсканировать $m$ страниц.

3. Для всех $i$: $s(i) := s(i) + \sum_{j=1}^{m} X(i, j)$, $t(i) := m$.

4. Для всех $i$: $\overline{X}(i) := s(i)/t(i)$ ($\overline{X}(i)$ – промежуточные средние значения количества ссылок, собранных со страницы сайта).

5. $n := \sum_{i=1}^{K} t(i)$, $X_{\max} := \max_{\substack{1 \le i \le K \\ 1 \le j \le t(i)}} X(i, j)$.

6. Найти $i^* := \arg\max_{1 \le i \le K} \left( \frac{\overline{X}(i)}{X_{\max}} + \sqrt{\frac{2 \ln(n)}{t(i)}} \right)$.

7. $t(i^*) := t(i^*) + 1$.

8. Сканировать страницу $t(i^*)$ сайта $i^*$.

9. $s(i^*) := s(i^*) + X(i^*, t(i^*))$, $\overline{X}(i^*) := s(i^*)/t(i^*)$, $n := n + 1$.

10. Если $n \le N$, то перейти к п. 5.
11. $S^{R2}(N) := \sum_{i=1}^{K} \sum_{j=1}^{t(i)} X(i,j)$.
12. Стоп.

## 4. Некоторые результаты

Для проведения эксперимента с целью апробации и сравнения трех решающих правил $R_1$, $R_2$ и $R_3$ было взято 10 официальных веб-сайтов институтов Санкт-Петербургского научного центра РАН и 10 официальных веб-сайтов высших учебных заведений Санкт-Петербурга. Веб-сайты сканировались в период с 20 по 25 ноября 2011 г. на глубину до 5-го уровня включительно, общее количество обработанных *html*-страниц равняется 105711, а общее количество собранных внешних гиперссылок – 8990. В таблице 1 приведены сведения о некоторых сайтах (так называемое «выборочное множество», содержащее 5 научных и 5 вузовских сайтов), а в таблице 2 – итоговые результаты эксперимента.

В частности, из таблицы 1 можно увидеть, что веб-сайты существенно различаются по количеству *html*-страниц и внешних гиперссылок, причём это различие характерно как для научных, так и для вузовских сайтов.

*Таблица 1. Выборочное множество*

| Название | Веб-сайт | Кол-во *html*-стр. | Кол-во внешних ссылок |
|---|---|---|---|
| Ботанический институт РАН | www.binran.ru | 1298 | 29 |
| Зоологический институт РАН | www.zin.ru | 1100 | 69 |
| Институт восточных рукописей РАН | www.orientalstudies.ru | 2411 | 48 |

*Таблица 1* (*продолжение*)

| | | | |
|---|---|---|---|
| Институт русской литературы РАН (Пушкинский дом) | www.pushkinskijdom.ru | 39292 | 3192 |
| Институт лингвистических исследований РАН | www.iling.spb.ru | 3523 | 129 |
| Петербургский госуниверситет путей сообщения | www.pgups.ru | 431 | 19 |
| Российский государственный гидрометеорологический университет | www.rshu.ru | 1766 | 409 |
| Санкт-Петербургский государственный политехнический университет | www.spbstu.ru | 1471 | 214 |
| Санкт-Петербургский госуниверситет | www.spbu.ru | 12935 | 903 |
| Санкт-Петербургский госуниверситет экономики и финансов | www.finec.ru | 7253 | 1048 |

*Таблица 2. Результаты эксперимента*

| Множество | Количество сайтов | $N$ | $S^{R1}$ | $S^{R2}$ | $S^{R3}$ |
|---|---|---|---|---|---|
| Институты РАН | 10 | 2000 | 417 | 816 | 457 |
| Вузы | 10 | 2000 | 1337 | 774 | 1447 |
| Выборочное множество | 10 | 2000 | 949 | 994 | 1078 |

## *5. Выводы и заключение*

Следует сказать, что как вузовские, так и научные сайты в целом характеризуются достаточно большим разбросом значений как количества страниц, так и количества исходящих гиперссылок. При этом характер расположения внешних ссылок на страницах весьма различен: для официальных вузовских сайтов характерно небольшое количество гиперссылок, сделанных с большого количества страниц, а для официальных сайтов научных учреждений РАН сайтов отмечается большое количество страниц, не имеющих внешних гиперссылок вообще, и небольшое количество страниц, имеющих очень много ссылок. Сказанное может служить объяснением плохого результата алгоритма $R_2$, основанного на индексах Гиттинса, применительно к вузовским сайтам. Этот алгоритм выбирает самые «привлекательные» сайты, и краулер сканирует их достаточно долго. Если же на первых страницах сайта очень мало внешних ссылок, то очередь до него дойдёт нескоро. Более того, если на первых $n$ страницах внешних ссылок нет, то краулер не дойдет до этого сайта никогда. Если за базовый результат принять количество гиперссылок, найденных по варианту $R_1$, то в случае научных институтов мы получаем почти двукратный рост найденных гиперссылок при использовании варианта $R_2$, а для вузов – рост 10% при использовании варианта $R_3$. Для выборочного множества также предпочтительнее вариант $R_3$.

Полученные результаты используются в модуле-планировщике блока управления *BeeBot* следующим образом: если пользователь указывает, что сканируемое множество является множеством официальных вузовских сайтов, то используется правило $R_3$, а для официальных сайтов РАН используется правило $R_2$. Если пользователь не указывает информации о целевом множестве, то каждому сайту целевого множества последовательно выделяется определенный квант времени для его сканирования.

Таким образом, в статье описан адаптивный краулер, предназначенный для сбора исходящих гиперссылок с заданного целевого множества веб-сайтов. Адаптивное поведение крауле-

ра, заключающееся в том, что за ограниченное время он должен находить максимально возможное суммарное количество гиперссылок, моделируется как задача о многоруком бандите. Такая постановка задачи позволила применить для её решения известные алгоритмы, эксперименты с которыми показали их зависимость от тематики целевого множества.

Очевидно, что задача разработка «хороших» адаптивных алгоритмов рационального поведения краулера имеет много направлений для развития, начиная от эвристических подходов до статистического анализа размещения внешних гиперссылок на сайтах заданной тематики.

### *Литература*

1. ВОРОНИН А.В., ПЕЧНИКОВ А.А. *Исследования сайтов органов власти Республики Карелия* // Век качества. – 2010. – №3. – С. 28–30.
2. ПЕЧНИКОВ А.А. *Методы исследования регламентируемых тематических фрагментов Web* // Труды Института системного анализа Российской академии наук. Серия: Прикладные проблемы управления макросистемами. – 2010. – Том 59. – С. 134–145.
3. ПЕЧНИКОВ А.А. *Модель университетского Веба* // Вестник Нижегородского университета им. Н.И. Лобачевского. – 2010. – №6. – С. 208–214.
4. ПЕЧНИКОВ А.А. *Исследование взаимосвязей между вебсайтами научных библиотек университетов России* // Дистанционное и виртуальное обучение. – 2011. – №7. – С. 13–24.
5. СОВЕТОВ Б.Я., ЯКОВЛЕВ С.А. *Моделирование систем*. – М.: Высшая школа, 2001. – 344 с.
6. AUER P., CESA-BIANCHI N., FISHER P. *Finite-time Analysis of the Multiarmed Bandit Problem* // Machine Learning. – 2002. – №47. – С. 235–256.
7. FIELDING R., GETTYS J., MOGUL J., NIELSEN H., MASINTER L., LEACH P., BERNERS-LEE T. *RFC 2616: Hypertext Transfer Protocol – HTTP/1.1. June 1999.* – URL: http://www.ietf.org/rfc/rfc2616.txt (07.12.2011).

8. MAHAJAN A., TENEKETZIS T. *Multi-armed bandit problems* // In Foundations and Applications of Sensor Management. – A. Hero, D. Castanon, D. Cochran, K. Rastella, eds. – Springer, 2008. – P. 121–151.
9. PANT G., SRINIVASAN P., MENCZER F. *Crawling the Web* // In "Web Dynamics". – M. Levene, Poulovassilis, eds. – Springer, 2004. – P.153–178.
10. SANG HO LEE, SUNG JIN KIM *On URL Normalization* // Lecture Notes in Computer Science. – 2005. – Vol. 47. – P. 1076–1085.
11. TACKSEUNG J. *A survey on the bandit problem with switching costs* // De Economist. – 2004. – Vol. 152. – P. 513–541.
12. THELWALL M. *Link Analysis: An Information Science Approach*. – Amsterdam: Elsevier Academic Press, 2004. – 269 p.

## ADAPTIVE CRAWLER FOR EXTERNAL HYPERLINKS SEARCH AND ACQUISITION

**Andrey Pechnikov**, Institute of Applied Mathematical Research Karelian Research Center of RAS, Petrozavodsk, Doctor of Science, assistant professor (pechnikov@krc.karelia.ru).
**Denis Chernobrovkin**, Faculty of applied mathematics and control processes Saint-Petersburg State University, graduate, Saint-Petersburg (8ezhikov@gmail.com).

*Abstract: We consider the web-crawler designed to find information about outgoing hyperlinks from a set of monothematic web-sites. Adaptive behavior of the crawler is formulated in terms of the multi-armed bandit problem. The experiments show that the choice of the adaptive algorithm of rational behavior depends on the subject of the considered set of web-sites.*

Keywords: hyperlink, crawler, adaptive behavior, multi-armed bandit problem, Gittins index.

*Статья представлена к публикации членом редакционной коллегии В. В. Мазаловым*