УДК 519.688

Е. В. Р т и щ е в а

АЛГОРИТМ ФОРМИРОВАНИЯ НАВИГАЦИОННОЙ СИСТЕМЫ ВЕБ-САЙТА

Проанализирована навигационная система веб-сайта. Описан метод построения семантической сети предметной области, представленной на сайте. Выявлены проблемы построения оптимизированной структуры навигации веб-сайта. Разработаны и описаны алгоритмы построения семантической сети предметной области и алгоритмы преобразования ее в систему навигации веб-сайта.

Интернет является навигационной системой, основным взаимодействием пользователя с которой является переход по гиперссылкам для перемещения в информационном пространстве сайта или сайтов. Из-за огромных размеров этого пространства навигация достаточно сложна. Интерфейс должен помогать пользователю отвечать на четыре вопроса [1]: где я нахожусь, где я уже побывал, куда мне идти дальше [1, 2], близка ли моя цель [3]?

Большинство сайтов имеет иерархическую структуру, или, по крайней мере, сайт можно разделить на подсайты, которые имеют иерархическую структуру, т.е. существует главный раздел (подраздел) сайта (подсайта), разделы меню которого ведут на более низкий уровень иерархии [1].

Алгоритм действий пользователя при посещении сайта следующий. Пользователь попадает на сайт (не обязательно на главную страницу, он может прийти и из поисковой системы). Далее он решает, куда перемещаться дальше, т.е. выбирает ссылку, которая наилучшим образом подходит для его целей. Если он не находит такой ссылки, то может вообще покинуть сайт.

Владельцы сайта заинтересованы в том, чтобы посетитель остался на сайте, нашел интересующую его информацию и приобрел товар или услугу. Для этого нужно наглядно представить систему навигации, что и делается с помощью правильно спроектированной структуры сайта: навигационного меню, его внешнего оформления, размещения на странице, дополнительного меню и т.д. [1, 2].

Большинство этих задач решается разработчиком сайта самостоятельно без какой-либо автоматизации, т.е. разработчик создает структуру сайта, руководствуясь собственным пониманием, а также различными рекомендациями.

В настоящее время почти все сайты имеют систему управления. Рассмотрим возможность частичной автоматизации построения системы навигации, но без учета ее расположения на странице и цветовых схем ее выделения.

Для того чтобы построить систему навигации, нужно: 1) построить семантическую сеть понятий предметной области; 2) преобразовать семантическую сеть до графа системы навигации сайта; 3) построить множество страниц и связей между ними.

Общее описание навигации по сайту. Главная страница сайта содержит главное меню, которое должно представлять собой *самый общий* уровень иерархии. Пунктами главного меню являются понятия предметной области, выявляемые на этапе проектирования. Каждому пункту меню соответствует отдельная страница, раскрывающая данный пункт. На такой странице есть свое меню, по которому можно перейти на более низкий уровень иерархии, раскрывающий это понятие глубже (рис. 1).

Представим веб-сайт в виде ориентированного размеченного графа, с одной страницы которого можно перейти на несколько страниц. Метки графа являются текстами ссылок. Пользователь видит только название пунктов меню, при инициализации которых активизируется ссылка. В зависимости от меток происходит переход на определенную вершину графа (рис. 2).

Построение семантической сети понятий предметной области. При проектировании сайта сначала задается его предметная область,

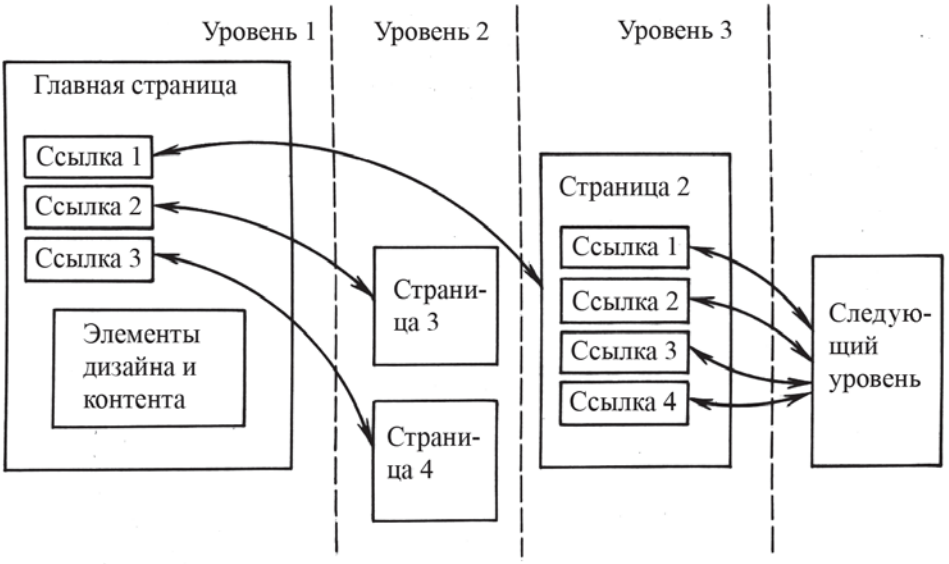

Рис. 1. Иерархическая структура сайта

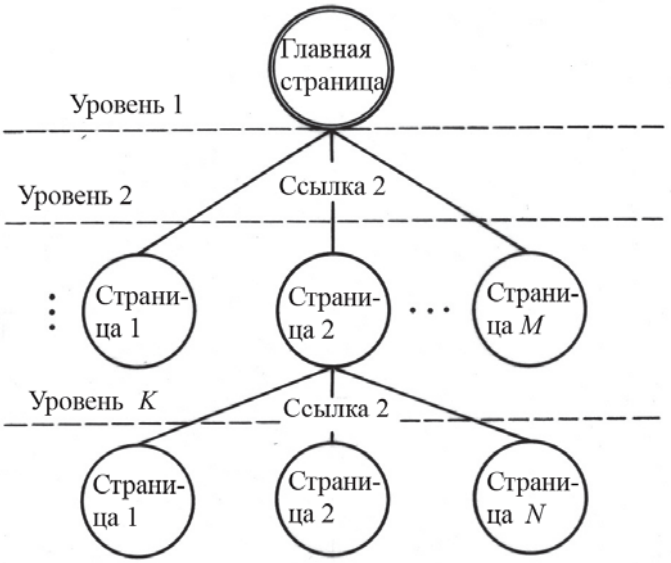

Рис. 2. Пример размеченного графа навигации сайта

выделяются понятия предметной области, которые должны быть представлены на сайте. Будем называть понятие предметной области *информационной единицей*. Множество понятий, имеющих одну степень обобщенности, будем называть *уровнем информации*. Степень обобщенности определяется разработчиком.

Для построения *семантической сети информационных единиц* все множество понятий предметной области нужно структурировать от общего к частному, а также провести типизацию связей данного понятия с другими понятиями, используя при этом стандартное множество семантических связей, например: родовидовая, часть–целое, ассоциативная, каузальная и др. [4].

Это делается следующим образом. 1. Выделяются понятия предметной области. Понятия предметной области и связи между ними *определяются разработчиком.*

2. Структурируется информация верхнего уровня. Верхний уровень (главная страница) должен представлять собой наиболее общую информацию; в данном случае под информацией понимаются ссылки меню, т.е. из всей массы информационных единиц разработчик выделяет самые главные [5].

3. Структурируется информация второго уровня. Для каждого из оставшихся элементов информации верхнего уровня выбираются элементы, являющиеся его подмножеством, они становятся пунктами подменю данного элемента. Определяются связи этих элементов между собой.

4. Структурируется информация третьего и последующих уровней. Для каждого элемента подменю выбираются элементы, являющиеся

его подмножеством. Это продолжается до тех пор, пока на множестве всех элементов не будет построен ориентированный граф — *ориентированная сеть*. Полученная сеть называется *семантической сетью информационных единиц*.

Важно понять, насколько можно автоматизировать построение семантической сети в соответствии с данным алгоритмом. Предположим, что разработчиком сайта уже спроектирована семантическая сеть. Теперь необходимо автоматизировать ее структурированный ввод в базу данных с определением иерархических уровней и связей между понятиями. Другой вариант — осуществляется выборка из некоторой базы данных понятий предметной области и связей между ними, эта база данных формируется экспертом в данной предметной области, что позволяет автоматизировать процесс построения сети. Но этот способ является более дорогостоящим и трудоемким, кроме того, проектировщик сайта обеспечивает более точный подбор понятий и связей между ними.

Алгоритм построения семантической сети. Пусть существует множество информационных единиц (понятий) предметной области данного сайта $I = (I_1, \ldots, I_m)$, где m — число информационных единиц, и существует некое множество семантических связей между этими ресурсами L, т.е. существует граф $G = (I, L)$, $L = (I_i, I_l)$, где $i \neq l$. Нужно найти топологию этого графа, т.е. структурировать информацию.

1. Множество ребер L искомого графа полагаем пустым: $L = \varnothing$.

2. Если $I \neq \varnothing$, то формируем множество понятий одного уровня: $I_T = \{\{I_{T_1}\}, \ldots, \{I_{T_n}\}\}$, где n — число информационных единиц этого уровня, а каждый элемент множества представляет собой множество, состоящее из одного элемента. Иначе переходим к седьмому шагу.

3. Если множество I_T содержит больше одного элемента, то переходим на к четвертому шагу, иначе — ко второму шагу.

4. Для каждого понятия I_{T_i} из множества $I \backslash I_T$ извлекаем понятие I_k.

5. Если $I_k \in I_{T_i}$, то переходим на шестой шаг, иначе — на четвертый.

6. Добавляем I_k в I_{T_i} (таким образом формируем множество дочерних понятий для I_{T_i}), убираем его из множества $I \backslash I_T$.

7. Переходим на пятый шаг, если для $I_{T_i} \exists\, I_k \in I_{T_i}$; если для $I_{T_i} \neg \exists\, I_k \in I_{T_i}$, то переходим к третьему шагу.

8. Прекращение работы. Множество L есть множество ребер искомого графа.

Получаем ориентированный граф вида $G(I, L)$, где I — множество информационных единиц, а L — множество связей между ними.

Пример фрагмента семантической сети для предметной области “Мавзолей В.И. Ленина” приведен на рис. 3, *а*. Видно, что нужно выделить уровни иерархии понятий, корнем дерева понятий является вершина “Ленин”, т.е. разработчик первоначально поставил понятия “Похороны Ленина” и “Ленин” на один уровень обобщения, но в процессе проектирования понятие “Похороны Ленина” перешло на третий уровень иерархии. Поэтому, чтобы обеспечить наглядность семантической сети для дальнейшей работы с ней, нужно применить следующий алгоритм.

Алгоритм выделения уровней иерархии понятий. Нужно упорядочить вершины графа так, чтобы выделить уровни иерархии понятий, т.е. понятия, имеющие одинаковую степень обобщенности, должны

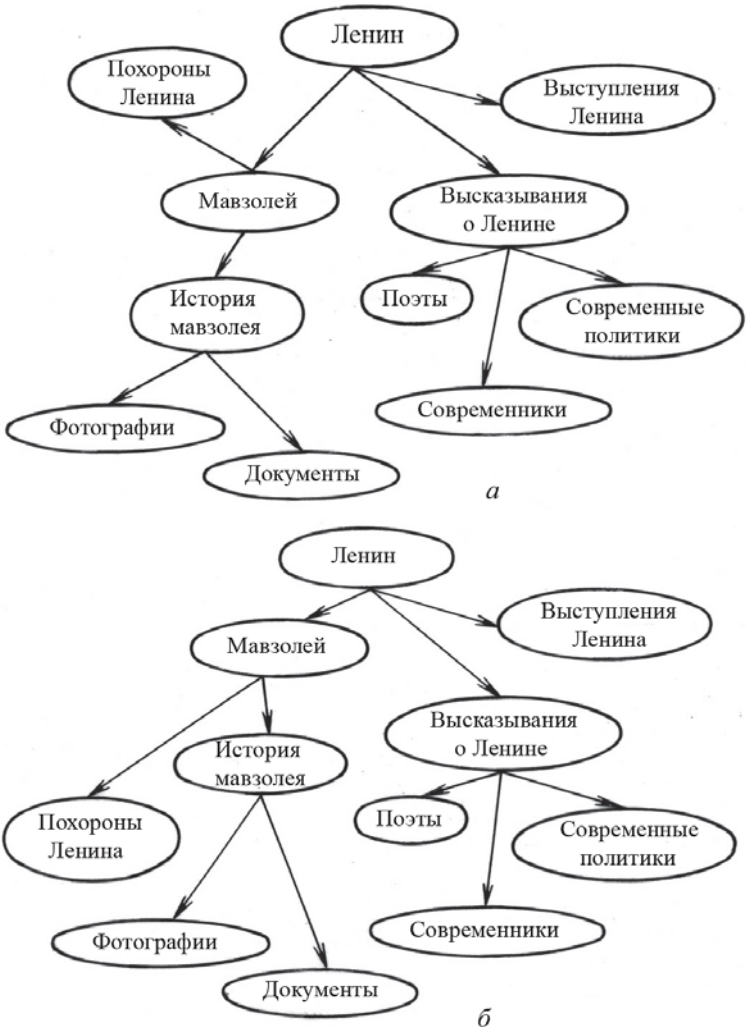

Рис. 3. Фрагмент семантической сети предметной области “Мавзолей В.И. Ленина” до (*а*) и после (*б*) выделения уровней иерархии понятий

располагаться на одном уровне, чтобы отчетливо было видно распределение этих вершин по уровням [6] для дальнейшей работы с ними. Для вычисления порядковой функции ориентированной сети используется *алгоритм Демукрона* [6].

Наглядно процесс определения уровней вершин можно представить следующим образом. Нулевой уровень образуют вершины с *полустепенью захода*, равной 0. Удалив из сети все вершины нулевого уровня и исходящие из них дуги, получим сеть, входами которой будут служить вершины первого уровня исходной сети. Этот процесс нужно продолжать, пока все вершины не будут распределены по уровням [6].

Алгоритм обрабатывает матрицу смежности B вершин графа порядка n. В результате получается массив Ord порядковых номеров уровней вершин.

0. Сформировать множество I вершин сети. Значение счетчика уровней $LN = 0$. Найти суммы элементов матрицы по всем столбцам матрицы B и заполнить ими массив M.

1. Если $I \neq \varnothing$, перейти на второй шаг, иначе — к третьему шагу.

2. Определить все вершины с полустепенью захода, равной 0, т.е. $M = 0$; это будет множество N.

3. Присвоить элементам массива Ord номера вершин из множества N, удалить вершины с этими номерами из множества I.

4. Вычесть из массива M строки матрицы B, соответствующие вершинам с номерами из множества N (т.е. вершинам из последнего вычисленного уровня).

5. Увеличить счетчик уровней на единицу: $LN = LN + 1$.

6. Вернуться к первому шагу.

7. Закончить работу. В результате получено множество $Ord = (N_1, \ldots, N_{LN})$, где LN — число уровней, N — множество номеров вершин данного уровня.

После применения алгоритма фрагмент сети имеет вид, приведенный на рис. 3, *б*.

Преобразование семантической сети информационных ресурсов в граф системы навигации сайта. По полученной семантической сети можно построить *граф системы навигации сайта.* Но этот граф может не отвечать требованиям оптимальной навигации сайта. Поэтому следует произвести дельнейшие преобразования. *Граф системы навигации сайта* — это граф, вершинами которого являются страницы с навигационной панелью (меню) или отдельными ссылками, по которым можно переходить на другие страницы (вершины графа). Если по этому графу строить навигацию сайта, то на каждой странице будет список ссылок для перехода на следующие страницы и т.д. Оптимальная структура навигации веб-сайта должна отвечать следующим требованиям.

Баланс глубины навигации. Существует мнение, что оптимальное число пунктов меню на странице должно быть не больше 7. . . 10 [5, 7, 8], в любом случае нужно ограничиться этим числом, но это, в свою очередь, увеличит число иерархических уровней сайта. В данном случае нужно придерживаться определенного баланса глубины и ширины навигации в зависимости от назначения сайта. Например, если число пунктов меню верхнего уровня составляет более 10 элементов, то их можно еще раз разбить на общие подгруппы, сгруппировать, выделив цветом, соответствующим образом озаглавить (как в каталогах ресурсов) и т.д. Либо увеличить размерность в глубину.

Баланс широты навигации. Если сайт содержит слишком много уровней (оптимальное число три), то можно на первый уровень вынести информацию второго уровня, сгруппировав ее соответствующим образом [9].

Далее будут описаны алгоритмы преобразования этой сети. В данном случае также невозможно достичь полной автоматизации, т.е. разработчик сам решает, какова будет глубина и широта навигации, и в соответствии с этим прерывает действие алгоритма там, где это ему нужно. Считается нецелесообразным жесткое задание широты и глубины навигации, так как правило “семь плюс-минус два” и правило “трех кликов” работают далеко не всегда. Если нужно убрать перекрестные ссылки, то где их убирать, решает сам разработчик.

В таблице приведены требования к графу информационных ресурсов и алгоритмы, с помощью которых преобразуется сеть для выполнения каждого требования.

Таблица

Требования к графу информационных ресурсов и алгоритмы, которые нужно применить для их выполнения

Требование	Алгоритм
Подразделы сайта не должны иметь перекрестных связей друг с другом	Алгоритм удаления перекрестных ссылок
Изменить глубину и ширину информационного графа	• Алгоритм объединения для уменьшения глубины навигации • Алгоритм объединения для уменьшения широты навигации
Вынести какой-либо ресурс на верхний уровень иерархии, чтобы упростить к нему доступ, если ресурс очень важен	Алгоритм объединения вершин графа в зависимости от важности ресурса

Алгоритм удаления перекрестных ссылок. Пусть существует множество вершин сети (I_{Level_0}), принадлежащих множеству $Level_0$ (полустепень захода равна нулю), и существует путь ненулевой длины из каждой вершины нулевого уровня в вершину самого нижнего уровня. Тогда *семантическим подразделом сети* является подграф, имею-

щий одну вершину с полустепенью захода, равной нулю, из которой достижимы все остальные вершины.

Пусть существует вершина I_i с полустепенью исхода, не равной нулю. Тогда *семантическим j-м подразделом сети* называется множество вершин, достижимых из вершины I_i. Семантические подразделы называются *пересекающимися*, если существует вершина, принадлежащая обоим подразделам. Эта вершина называется *общей*.

Множество вершин из уровня i, связанных отношением непосредственной достижимости с вершиной из уровня $i+1$, называется *множеством родительских вершин*.

Множество вершин из уровня $i+1$, связанных отношением непосредственной достижимости с вершиной из уровня i, называется *множеством дочерних вершин.*

Если нужно, чтобы какие-либо подразделы сети не имели общих вершин, то используется этот алгоритм. Определяется вершина с полустепенью исхода, не равной нулю (это может быть вершина из нулевого уровня), относительно нее строится семантический подраздел сети. Берется еще одна вершина, принадлежащая тому же уровню, и относительно нее также строится семантический подраздел сети, и так поступают со всеми вершинами данного уровня. Далее ищутся общие вершины. Каждая такая вершина анализируется, так же анализируется ее связь с родительскими вершинами. В зависимости от решения разработчика происходит следующие: удаляется одна дуга; вершина делится на две; ничего не меняется и осуществляется переход к следующей вершине.

Этот алгоритм применим для всех других уровней[1].

0. Составить матрицу смежности B, посчитать сумму элементов по всем столбцам, заполнить ими массив M.

1. Определить все вершины с полустепенью захода, большей единицы, т.е. $M_i > 1$; это будет множество I_{cross}.

2. Для каждой вершины множества I_{cross} анализируются связи с родительскими вершинами.

3. Из множества дуг $L = ((I_{p1}, I_{i_{cross}}), \ldots, (I_{pm}, I_{i_{cross}}))$ удаляются ненужные дуги, а из матрицы M — элементы, соответствующие этим дугам.

4. Если нужно еще делить вершины на две, то переходим к пятому шагу, иначе — к девятому шагу.

5. Определяем все вершины с полустепенью захода, большей единицы, т.е. $M_i > 1$; это будет множество I_{cross}.

[1] Для упрощения алгоритма будем считать, что вершины можно делить только на две части, а если одну из полученных вершин нужно разделить еще раз, то для этого надо повторить алгоритм разделения.

6. Если $I_{cross} \neq \varnothing$, то переходим на седьмой шаг, иначе — на девятый.

7. Делим вершины на две, добавляем в матрицу смежности строку и столбец, соответствующие этой вершине.

8. Если нужно еще делить вершины на две, то переходим к пятому шагу, иначе — к девятому шагу.

9. Завершение работы: получение новой матрицы смежности M и графа $G = (I, L)$.

Алгоритм объединения для уменьшения глубины навигации. Если степень глубины навигации неудовлетворительна, то можно использовать следующий алгоритм.

Существует граф информационных ресурсов $G = (I, L)$. Существуют две вершины I_i, I_k и дуга (I_i, I_k), т.е. I_k принадлежит более низкому уровню иерархии, чем I_i. Если мы хотим уменьшить степень глубины навигации или число узлов (понятий) на одном уровне, то эти вершины можно объединить: $Level_j$ — множество вершин j-го уровня иерархии, $j = (1, \ldots, LN)$.

1. Существует граф $G(I, L)$, определяем уровень, который будем объединять с последующим.

2. Определяем множество дуг (I_i, I_k), где $I_i \in Level_j$, а $I_k \in Level_{j+1}$.

3. Объединяем попарно вершины (если необходимо).

4. Если нужно объединять еще, то переходим ко второму шагу, если не нужно — к пятому шагу.

5. Завершение работы: получение оптимизированного графа $G = (I, L)$.

Алгоритм объединения для уменьшения широты навигации. Если число понятий (пунктов меню) на одном уровне слишком большое, то можно использовать следующий алгоритм.

Существует граф информационных ресурсов $G = (I, L)$. Существует множество вершин, принадлежащих j-му уровню иерархии, имеющих одну и ту же *родительскую вершину* (т.е. имеющих полустепень захода, равную единице). Если мы хотим уменьшить число узлов (понятий) на одном уровне, то эти вершины можно объединить: $Level_j$ — множество вершин j-го уровня иерархии, $j = (1, \ldots, LN)$.

1. Существует граф $G(I, L)$, определяем уровень, число узлов на котором нужно сократить.

2. Определяем $I_i \in Level_j$ и множество вершин $(I_k) \in Level_{j+1}$, где полустепень захода равна единице и все вершины из множества (I_k) связаны отношением непосредственной достижимости с I_i.

3. Объединяем вершины (если необходимо).

4. Если нужно еще объединить вершины, переходим на второй, если не нужно — на пятый шаг.

5. Завершение работы: получение оптимизированного графа $G = (I, L)$.

Алгоритм объединения вершин графа в зависимости от важности ресурса. Если какой-либо ресурс особенно важен для посетителя сайта, то его можно вынести в предыдущий уровень иерархии, несмотря на то что в результате действия предыдущих алгоритмов он оказался внизу. Кроме того, если какой-либо ресурс не очень важен, то его можно опустить на нижний уровень иерархии.

Дуги семантической сети информационных ресурсов размечаются в зависимости от важности ресурса для пользователя, т.е. получаем граф $G = (I, L)$, где вес ребра e обозначим $\varphi(e)$. Нужно найти такие пути от начальной вершины до заданной, где $\sum_{e \in G} \varphi(e) \geqslant \max$ ($\max$ — максимальное значение стоимости пути в графе). В результате две соседние вершины объединяются в одну, это нужно для ускорения доступа пользователя к важной информации.

1. Устанавливаем $M = \max$.

2. Если множество вершин j-го уровня $Level_j \neq \varnothing$ и $Level_{j+1} \neq \varnothing$, то переходим к третьему шагу, иначе — к шестому шагу.

3. Берем I_i из $Level_j$.

4. Если $\exists \varphi(e) \geqslant M$, где $e = \{I_{i_{Level_j}}, I_{i_{Level_{j+1}}}\}$, то переходим на пятый шаг, иначе — на второй.

5. Объединяем $I_{i_{Level_j}}$ с $I_{i_{Level_{j+1}}}$, переходим ко второму шагу.

6. Если множество $L \neq \varnothing$, то увеличиваем номер подуровня на единицу, т.е. $j = j + 1$, иначе переходим к седьмому шагу.

7. Завершение работы: получение оптимизированного графа $G = (I, L)$.

Вывод. Разработанные алгоритмы можно применять при построении графа навигации сайта. Они просты в реализации, не требуют большого количества ресурсов.

СПИСОК ЛИТЕРАТУРЫ

1. Нильсен Я. WEB-дизайн. – СПб.: Символ, 2003. – 512 с.
2. Пауэлл Т. А. Web-дизайн. – СПб.: БХВ-Петербург, 2004. – 1072 с.
3. Джонсон Дж. Web-дизайн: типичные ляпы и как их избежать. – М.: Кудиц-образ, 2005. – 400 с.
4. Терно Ю. А. Построение информационной модели WEB-сайта на основе семантического структурирования. http://joker.botik.ru/show_thesis.php3?year=2000&name=terno.
5. Мельников М. С чего начинается сайт http://cherry-design.spb.ru/articles/article_28.htm.
6. Белоусов А. И. Ткачев С. Б. Дискретная математика. – М.: Изд-во МГТУ им. Н.Э. Баумана, 2002. – 744 с.

7. Ф й е А. Рентабельный web-дизайн. – М.: Кудиц-образ, 2004. – 336 с.
8. К и р с а н о в Д. WEB-дизайн. – СПб.: Символ, 2003. – 368 с.
9. K a l b a c h J. The Myth of “Seven, Plus or Minus 2” http://www.ddj.com/184412300.

Статья поступила в редакцию 15.02.2007

Елена Валерьевна Ртищева родилась в 1979 г., окончила МГТУ им. Н.Э. Баумана в 2003 г. Аспирант кафедры “Системы обработки информации и управления” МГТУ им. Н.Э. Баумана. Автор шести научных работ в области проектирования программных систем. Специализируется в области проектирования веб-сайтов и систем управления веб-сайтами.

Ye. V. Rtishcheva (b. 1979) graduated from the Bauman Moscow State Technical University in 2003. Post-graduate of “Systems of Data Processing and Control” department of the Bauman Moscow State Technical University. Author of 6 publications in the field of program systems design. Specializes in the field of web-site design and systems of web-site management.

УДК 621.396

М. К. Ч о б а н у, А. В. С е р ж а н т о в, К. З а н д е р

ВЫБОР ОПТИМАЛЬНОЙ ЦВЕТОВОЙ МОДЕЛИ ИЗОБРАЖЕНИЯ В ЦЕЛЯХ ЕГО КОДИРОВАНИЯ ИЕРАРХИЧЕСКИМ АЛГОРИТМОМ

Рассмотрен процесс предобработки цветных изображений для последующего их сжатия иерархическим алгоритмом кодирования. Проведена классификация методов обработки, обоснован выбор оптимального из них на основе требований алгоритма сжатия к входным данным. Приведены результаты тестирования выбранных методов, а также предложены некоторые модификации, улучшающие основные показатели предобработки — качество и быстродействие.

В связи с широким распространением цифровых устройств (фотоаппаратов, видеокамер и др.) проблема хранения цветных фото- и видеоизображений становится все более актуальной, так как изображения и видеосигналы занимают огромные объемы памяти. Основными вопросами при решении данной проблемы являются предобработка данных и их сжатие. Под предобработкой цветного изображения понимается его преобразование в такую цветовую систему, которая описывала бы цвет, удовлетворяя всем требованиям алгоритма сжатия.

Выбор цветовой системы, описывающей цветное изображение определяется требованиями иерархического алгоритма, применяемого