УДК 57.087.1

Кудрина М.А., Кудрин К.А., Дегтярева О.А., Сопченко Е.В.

ФГАОУ ВО «Самарский государственный аэрокосмический университет им. академика С.П. Королева (Национальный исследовательский университет)», Самара, Россия

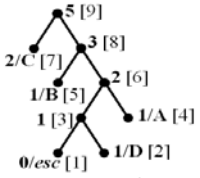
АДАПТИВНЫЙ АЛГОРИТМ ХАФФМАНА СЖАТИЯ ИНФОРМАЦИИ

Адаптивный алгоритм Хаффмана является модификацией обычного алгоритма Хаффмана сжатия сообщений. Он позволяет не передавать таблицу кодов и ограничиться одним проходом по сообщению, как при кодировании, так и при декодировании.

Суть адаптивного алгоритма состоит в том, что при каждом сопоставлении символу кода изменяется внутренний ход вычислений так, что в следующий раз этому же символу может быть сопоставлен другой код, т.е. происходит адаптация алгоритма к поступающим для кодирования символам [1]. При декодировании происходит аналогичный процесс.

В адаптивном алгоритме сжатия Хаффмана используется упорядоченное бинарное дерево. Бинарное дерево называется *упорядоченным*, если его узлы могут быть перечислены в порядке неубывания веса. Перечисление узлов происходит по ярусам снизу-вверх и слева-направо в каждом ярусе. Узлы, имеющие общего родителя, находятся рядом на одном ярусе.

На рис. 1 приведен пример упорядоченного дерева Хаффмана. Рядом с узлами дерева на данном рисунке приведены их веса, а в квадратных скобках приведены порядковые номера узлов дерева при перечислении.



№ узла	1	2	3	4	5	6	7	8	9
Вес узла	0	1	1	1	1	2	2	3	5

Рисунок 1 - Пример упорядоченного дерева Хаффмана

Правила построения и упорядочивания бинарного дерева Хаффмана

В начале работы алгоритма дерево кодирования содержит только один специальный символ, всегда имеющий частоту 0. Он необходим для занесения в дерево новых символов. Обычно этот символ называют *escape*-символом (<*esc*>).

Левые ветви дерева помечаются 0, а правые – 1.

При нарушении упорядоченности дерева (после добавлении нового листа или изменении веса имеющегося листа) его необходимо упорядочить.

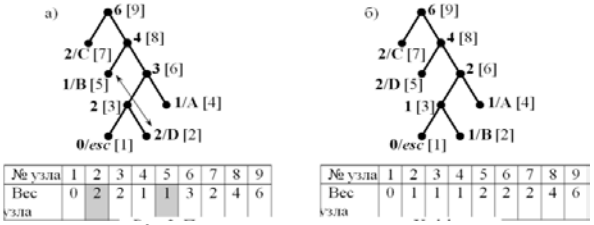
Чтобы упорядочить дерево необходимо поменять местами два узла: узел, вес которого нарушил упорядоченность, и последний из следующих за ним узлов меньшего веса. После перемены мест узлов необходимо пересчитать веса всех их узлов-предков.

Рассмотрим пример упорядочивания дерева Хаффмана. Возьмем в качестве исходного дерева дерево, представленное на рис.1. Пусть на вход поступил очередной символ D. Вес листа D увеличился и стал равен 2. На рис. 2а приведен порядок перечисления узлов и последовательность весов, которая не является неубывающей. Следовательно, двоичное дерево не упорядочено. Для упорядочивания необходимо поменять местами лист D (нарушивший упорядоченность) и лист B (последний из меньших по весу узлов при перечислении). Результат упорядочивания приведен на рис. 2б.

Правила кодирования

1. Элементы входного сообщения считываются побайтно.

2. Если входной символ присутствует в дереве, в выходной поток записывается код, соответствующий последовательности нулей и единиц, которыми помечены ветки дерева, при проходе от корня дерева к данному листу. Вес данного листа увеличивается на 1. Веса узлов-предков корректируются. Если дерево становится неупорядоченным - упорядочивается.



№ узла	1	2	3	4	5	6	7	8	9
Вес узла	0	2	2	1	1	3	2	4	6

№ узла	1	2	3	4	5	6	7	8	9
Вес узла	0	1	1	1	2	2	2	4	6

Рисунок 2 - Пример упорядочивания дерева Хаффмана

3. Если очередной символ, считанный из входного сообщения при сжатии, отсутствует в дереве, в выходной поток записывается набор нулей и единиц, которыми помечены ветки бинарного дерева при движении от корня к *escape*-символу, а затем 8 бит *ASCII*-кода нового символа. В дерево вместо *escape*-символа добавляется ветка: родитель, два потомка. Левый потомок становится *escape*-символом, правый - новым добавленным в дерево символом. Веса узлов-предков корректируются, а дерево при необходимости упорядочивается.

Например, если мы имеем дерево Хаффмана как на рис. 3а, и очередной символ на входе C, которого еще нет в дереве, то выходной код нового символа будет 00'C', где 'C' - *ASCII*-код символа C. В дерево добавляется ветка с новым символом (см. рис. 3б) и дерево упорядочивается (см. рис. 3в).

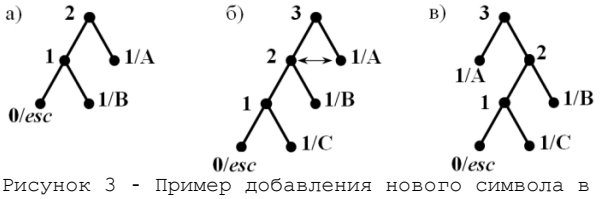


Рисунок 3 - Пример добавления нового символа в дерево Хаффмана

Рассмотрим пример работы алгоритма архивации на следующей входной последовательности символов: **ABCCDDDDBB**.

Вх. данные	Вых. данные	№ дерева	Длина кода, бит
A	'A'	1	8
B	0'B'	2	9
C	00'C'	3	10
C	101	4	3
D	100'D'	5	11
D	1101	6	4
D	10	7	2
D	0	8	1
B	1101	9	4
B	111	10	3

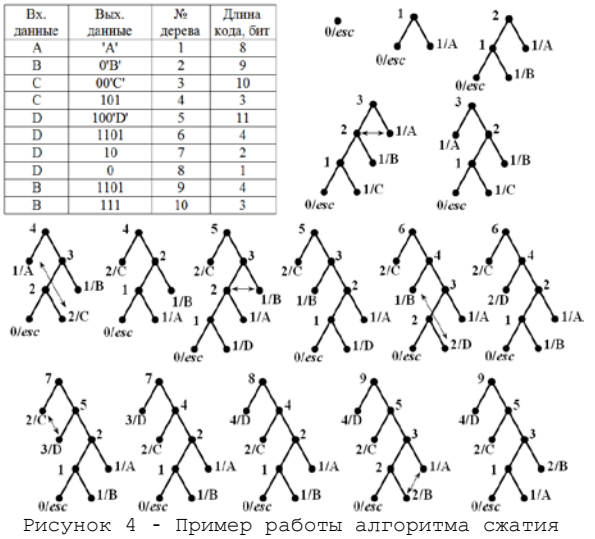


Рисунок 4 - Пример работы алгоритма сжатия

Сообщение из 10 символов занимает 80 бит. Сжатые данные: **'A'0'B'00'C'101100'D'11011001101111**. Сжатое сообщение занимает 55 бит.

Для сравнения подсчитаем, сколько бит займет данное сообщение, закодированное обычным методом Хаффмана. Таблица кодировки и кодовое дерево приведены на рис. 5.

Символ	Вероятность	Код
D	0,4	0
B	0,3	11
C	0,2	101
A	0,1	100

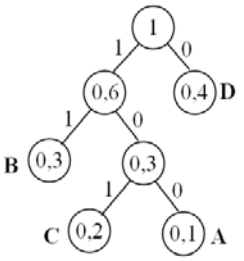


Рисунок 5 - Дерево Хаффмана

Закодированное сообщение: **1001110110100001111** (всего 19 бит). Кроме того, следует учесть, что вместе с закодированным сообщением передается и таблица кодировки. В данном случае это как минимум еще 4 байта *ASCII*-кодов символов плюс соответствующие им коды Хаффмана. Всего 68 бит.

Правила декодирования

1. Элементы входного сообщения считываются побитно.

2. Каждый раз при считывании 0 или 1 происходит перемещение от корня вниз по соответствующей ветке бинарного дерева Хаффмана, до тех пор, пока не будет достигнут какой-либо лист дерева.

3. Если достигнут лист, соответствующий символу, в выходное сообщение записывается *ASCII*-код данного символа. Вес листа увеличивается на 1, веса узлов-предков корректируются, дерево при необходимости упорядочивается.

4. Если же достигнут *escape*-символ, из входного сообщения считываются 8 следующих бит, соответствующих *ASCII*-коду нового символа. В выходное сообщение записывается *ASCII*-код данного символа. В дерево добавляется новый символ, веса узлов-предков корректируются, затем при необходимости производится его упорядочивание.

Рассмотрим процесс декодирования сообщения **'A'0'B'0100'C'11**. В начале декодирования дерево Хаффмана содержит только *escape*-символ с частотой 0. С раскодированием каждого нового символа дерево перестраивается (см. рис. 6). На выходе получим последовательность **ABBCBB**.

Вх. данные	Вых. данные	№ дерева
'A'	A	1
0'B'	B	2
01	B	3
00'C'	C	4
1	B	5
1	B	6

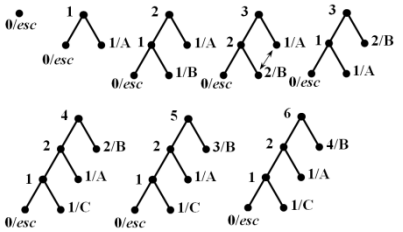


Рисунок 6 - Пример работы алгоритма декомпрессии

Программа демонстрации работы алгоритмов сжатия

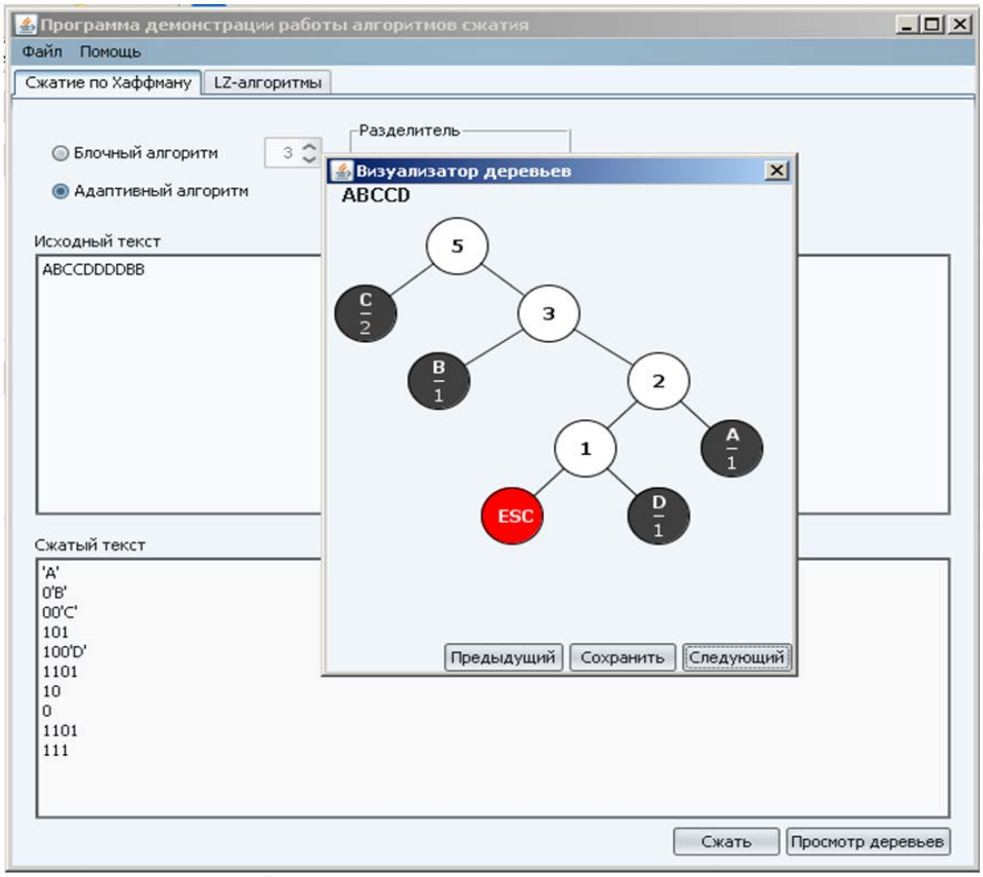


Рисунок 7 - Программа-визуализатор алгоритмов сжатия

Адаптивное кодирование Хаффмана имеет ряд особенностей при своей работе [2, 3]. Одна из них связана с хранением значений кодов символов, при кодировании длинных последовательностей. Дело в том, что языки высокого уровня, обычно, оперируют с числами максимум в 32 бит, в которых можно записывать числа от 0 до 4294967296, т.е. можно работать с файлами размером 4Гб. Но иногда этих значений недостаточно и приходится вырабатывать дополнительные меры по предотвращению переполнения счетчиков символов. Правильное решение может заключаться в накоплении битов кода в связанном списке, к которому можно добавлять новые узлы. Следующая проблема связана с определением конца передачи. Ведь для декомпрессора не существует ни одного способа определения конца передачи. Есть, по крайней мере, два способа решения этой проблемы. Компрессор может добавить к началу сжатых данных длину файла, делая их на пару байт длиннее. Либо можно в начале кодирования добавить в дерево помимо листа с *escape*-символом еще один вспомогательный лист с нулевой частотой, код которого сообщит декомпрессору о конце передачи данных.

Но, несмотря на указанные сложности, он часто применяется на практике, например, в протоколе *V*.32 передачи данных по модему со скоро-

стью 14400 бод или в программе *compact* операционной системы *UNIX*. Этот алгоритм наиболее эффективен для сжатия текстов или программных файлов. Изображения лучше сжимаются другими алгоритмами сжатия.

На кафедре информационных систем и технологий Самарского государственного аэрокосмического университета им. академика С.П. Королева разработана программа-визуализатор алгоритмов сжатия информации. Данная программа используется в учебном процессе в курсе "Теория информации" [4] при проведении лабораторных работ, а также для создания аттестационных педагогических измерительных материалов. Программа позволяет автоматизировать процесс составления тестовых задач. Также визуализатор демонстрирует студентам работу следующих алгоритмов сжатия: алгоритм Хаффмана с блокированием и без, адаптивный алгоритм Хаффмана, алгоритмы семейства *LZ* (*LZ77*, *LZ78*, *LZSS*). В будущем планируется расширить возможности программы путем добавления в нее других алгоритмов сжатия данных.

На рис. 7 приведен скриншот работы адаптивного алгоритма Хаффмана

ЛИТЕРАТУРА

1. *Лидовский В.В.* Теория информации. - М.:Наука, 2003. - 112 с.
2. http://old.reslib.com/book/Teoriya_informacii__Lidovskij_V_V__
3. http://rain.ifmo.ru/cat/view.php/theory/data-compression/adaptive-huffman-2006
4. http://sernam.ru/cod_2.php
5. Горячев Н.В. К вопросу реализации метода автоматизированного выбора системы охлаждения / Горячев Н.В., Кочегаров И.И., Юрков Н.К. // Алгоритмы, методы и системы обработки данных. 2013. № 3 (25). С. 16-20.
6. Трусов В.А. Однопозиционный модуль управления шаговым двигателем / Трусов В.А., Кочегаров И.И., Горячев Н.В., Юрков Н.К. // Теоретические и прикладные аспекты современной науки. 2015. № 7-3. С. 131-133.
7. *Кудрина М.А., Кудрин К.А., Попова-Коварцева Д.А.* Учебно-методический комплекс дисциплины «Теория информации» // Надежность и качество - 2012: труды Межд. симпозиума: в 2 т./ под ред. Н.К. Юркова. – Пенза: Изд-во ПГУ, 2012. – 1 т. С. 376-377.

УДК 621.92

Макаров В.Ф., Никитин С.П.

ФГБОУ ВПО «Пермский национальный исследовательский политехнический университет», Пермь, Россия

ОБЕСПЕЧЕНИЕ КАЧЕСТВА И УСТАЛОСТНОЙ ПРОЧНОСТИ ЛОПАТОК ТУРБИН НА ОСНОВЕ МОДЕЛИРОВАНИЯ ДИНАМИЧЕСКОЙ СИСТЕМЫ СТАНКА И ПРОЦЕССА ГЛУБИННОГО ШЛИФОВАНИЯ

Для изготовления рабочих и сопловых лопаток турбины газотурбинных двигателей (ГТД) используют жаропрочные литейные никелевые сплавы. Одним из основных методов обработки базовых поверхностей лопаток ГТД является глубинное профильное шлифование на многокоординатных станках.

При этом возникают проблемы с размерной точностью сложного профиля лопатки, а также с дефектами поверхностного слоя в виде прижогов и трещинообразования на ряде поверхностей [1], что снижает качество и предел выносливости σ_{-1} лопаток. Для обеспечения заданной размерной точности и качества поверхностного слоя при обработке лопаток ГТД приходится реализовывать множество проходов, что снижает производительность глубинного шлифования.

Для повышения производительности и эффективности глубинного шлифования при заданных параметрах качества необходимо прогнозирование качества деталей при обработке лопаток газотурбинных авиадвигателей, выявление взаимосвязи эксплуатационных показателей качества лопаток с параметрами поверхностного слоя и технологическими показателями процесса глубинного шлифования на основе математических моделей [1].

Процесс прогноза можно представить в виде трех этапов.

На первом этапе требуется установить зависимость колебаний сил и температур в зоне резания, шероховатости поверхности от технологических режимов и характеристики круга.

$$\begin{aligned} \theta &= f\left(v,s,t,K^0,a_3\ldots\right),\\ P &= f\left(v,s,t,K^0,a_3\ldots\right),\\ R_a &= f\left(v,s,t,K^0,a_3\ldots\right) \end{aligned} \qquad (1)$$

Далее необходимо выявить влияние температур в зоне резания и нагрузок на наклеп и остаточные напряжения в обрабатываемом материале [2, 3].

$$\begin{aligned} \sigma_{\text{ост}} &= f\left(\theta,P,R_a\right),\\ N &= f\left(\theta,P,R_a\right),\\ h_{\text{н}} &= f\left(\theta,P,R_a\right),\\ u_{\text{н}} &= f\left(\theta,P,R_a\right) \end{aligned} \qquad (2)$$

Затем следует определить влияние параметров поверхностного слоя на предел выносливости σ_{-1} лопаток.

$$\sigma_{-1} = f\left(\sigma_{\text{ост}},T,R_a\right) \qquad (3)$$

Такие математические зависимости, связывающие предел выносливости σ_{-1} с шероховатостью, наклепом и остаточными напряжениями были получены Т.В. Серенсеном, В.П. Когаевым, Р.Б. Шнейдеровичем [2].

Значительное внимание на первом этапе должно быть уделено исследованию динамики и устойчивости процесса, термодинамическим явлениям при профильном глубинном шлифовании [3,4]. Качество поверхностного слоя заготовки определяется одновременным воздействием геометрического, силового и теплового факторов, являющихся функциями процесса шлифования и имеющих непосредственную связь с относительными колебаниями формообразующих узлов станка [4]. Колебания при шлифовании являются вынужденными, но их уровень зависит от степени устойчивости динамической системы. Колебания динамической системы станка при шлифовании вызывает изменение фактического срезаемого слоя, фактической силы резания, а в силу взаимосвязанности процессов вызывают изменения в тепловой системе станка при шлифовании. Это приводит к колебаниям температур в зоне шлифования.

Задача заключается в поиске путей снижения колебаний шлифовальных станков за счет повышения степени устойчивости системы. Это позволит управлять тепловыми и упругими явлениями при шлифовании так, чтобы обеспечить заданное качество поверхностного слоя лопатки и предел выносливости. Поэтому при исследовании определяют способы исключения колебаний или понижения их уровня. В рамках решения задачи были проведены теоретические исследования влияния конструктивных параметров и режимов резания на устойчивость и уровень температур при глубинном шлифовании.

До настоящего времени, тепловые и динамические процессы при шлифовальной обработке исследовались раздельно. Но при использовании предельных режимов шлифования и повышении требований к точности обработки постоянные времени этих процессов в зоне резания становятся сравнимыми. Поэтому динамическое поведение техноло-