SOFTWARE
PRACTICE & EXPERIENCE

VOLUME 25, No. 10 OCTOBER 1995

EDITORS

DOUGLAS COMER

ANDY WELLINGS

WILEY
Publishers Since 1807
Chichester · New York · Brisbane · Toronto · Singapore
A Wiley–Interscience Publication

SPEXBL 25(10) 1065–1182 (1995)
ISSN 0038-0644

SOFTWARE
PRACTICE & EXPERIENCE

Aims and Scope

Software—Practice and Experience is an internationally respected and rigorously refereed vehicle for the dissemination and discussion of practical experience with new and established software for both systems and applications. Contributions regularly: (a) describe detailed accounts of completed software-system projects which can serve as 'how-to-do-it' models for future work in the same field; (b) present short reports on programming techniques that can be used in a wide variety of areas; (c) document new techniques and tools that aid in solving software construction problems; and (d) explain methods/techniques that cope with the special demands of large scale software projects. The journal also features timely Short Communications on rapidly developing new topics.

The editors actively encourage papers which result from practical experience with tools and methods developed and used in both academic and industrial environments. The aim is to encourage practitioners to share their experiences with design, implementation and evaluation of techniques and tools for software and software systems.

Papers cover software design and implementation, case studies describing the evolution of system and the thinking behind them, and critical appraisals of software systems. The journal has always welcomed tutorial articles describing well-tried techniques not previously documented in computing literature. The emphasis is on practical experience; articles with theoretical or mathematical content are included only in cases where an understanding of the theory will lead to better practical systems.

Articles range in length from a Short Communication (half to two pages) to the length required to give full treatment to a substantial piece of software (40 or more pages).

Software—Practice and Experience (ISSN 0038–0644/USPS 890-920) is published monthly, by John Wiley & Sons Limited, Baffins Lane, Chichester, Sussex, England. Second class postage paid at Jamaica, N.Y. 11431. Air freight and mailing in the U.S.A. by Publications Expediting Services Inc., 200 Meacham Avenue, Elmont, N.Y. 11003. © 1995 by John Wiley & Sons Ltd. Printed and bound in Great Britain by Page Bros, Norwich. Printed

SOFTWARE—PRACTICE AND EXPERIENCE
(Softw. pract. exp.)

CONTENTS

VOLUME 25, ISSUE No. 10 *October 1995*

SPEXBL 25(10) 1065–1182 (1995)
ISSN 0038–0644

Indexed or abstracted by Cambridge Scientific Abstracts, CompuMath Citation Index (ISI), Compuscience Database, Computer Contents, Computer Literature Index, Computing Reviews, Current Contents/Eng, Tech & Applied Sciences, Data Processing Digest, Deadline Newsletter, Educational Technology Abstracts, Engineering Index, Engineering Societies Library, IBZ (International Bibliography of Periodical Literature), Information Science Abstracts (Plenum), INSPEC, Knowledge Engineering Review, Nat Centre for Software Technology, Research Alert (ISI) and SCISEARCH Database (ISI).

SOFTWARE—PRACTICE AND EXPERIENCE, VOL. 25(10), 1097–1116 (OCTOBER 1995)

Automatic Synthesis of Compression Techniques for Heterogeneous Files

WILLIAM H. HSU

Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, IL 61801, U.S.A.
(email: bhsu@cs.uiuc.edu, voice: (217) 244-1620)

AND

AMY E. ZWARICO

Department of Computer Science, The Johns Hopkins University, Baltimore, MD 21218, U.S.A.
(email: amy@cs.jhu.edu, voice: (410) 516-5304)

SUMMARY

We present a compression technique for heterogeneous files, those files which contain multiple types of data such as text, images, binary, audio, or animation. The system uses statistical methods to determine the best algorithm to use in compressing each block of data in a file (possibly a different algorithm for each block). The file is then compressed by applying the appropriate algorithm to each block. We obtain better savings than possible by using a single algorithm for compressing the file. The implementation of a working version of this heterogeneous compressor is described, along with examples of its value toward improving compression both in theoretical and applied contexts. We compare our results with those obtained using four commercially available compression programs, PKZIP, Unix `compress`, *StuffIt*, and *Compact Pro*, and show that our system provides better space savings.

KEY WORDS: adaptive/selective data compression algorithms; redundancy metrics; heterogeneous files; program synthesis

INTRODUCTION

The primary motivation in studying compression is the savings in space that it provides. Many compression algorithms have been implemented, and with the advent of new hardware standards, more techniques are under development. Historically, research in data compression has been devoted to the development of algorithms that exploit various types of redundancy found in a file. The shortcoming of such algorithms is that they assume, often inaccurately, that files are homogeneous throughout. Consequently, each exploits only a subset of the redundancy found in the file.

Unfortunately, no algorithm is effective in compressing all files.[1] For example, dynamic Huffman coding works best on data files with a high variance in the frequency of individual characters (including some graphics and audio data), achieves mediocre performance on natural language text files, and performs poorly in general on high-redundancy binary data. On the other hand, run length encoding works well on high-redundancy binary data, but performs very poorly on text files. Textual substitution works best when multiple-character strings tend to be repeated, as in English text, but this performance degrades as the average

CCC 0038–0644/95/101097–20

Received 20 April 1994
Revised 5 February 1995

length of these strings decreases. These relative strengths and weaknesses become critical when attempting to compress *heterogeneous* files. Heterogeneous files are those which contain multiple types of data such as text, images, binary, audio, or animation. Consequently, their constituent parts may have different degrees of compressibility. Because most compression algorithms are either tailored to a few specific classes of data or are designed to handle a single type of data at a time, they are not suited to the compression of heterogeneous files. In attempting to apply a single method to such files, they forfeit the possibility of greater savings achievable by compressing various segments of the file with different methods.

To overcome this inherent weakness found in compression algorithms, we have developed a *heterogeneous compressor* that automatically chooses the best compression algorithm to use on a given variable-length block of a file, based on both the qualitative and quantitative properties of that segment. The compressor determines and then applies the selected algorithms to the blocks separately. Assembling compression procedures to create a specifically tailored program for each file gives improved performance over using one program for all files. This system produces better compression results than four commonly available compression packages, PKZIP,[2] Unix `compress`,[3] *StuffIt*,[4] and *Compact Pro*[5] for arbitrary heterogeneous files.

The major contributions of this work are twofold. The first is an improved compression system for heterogeneous files. The second is the development of a method of statistical analysis of the compressibility of a file (its redundancy types). Although the concept of redundancy types is not new,[6,7] synthesis of compression techniques using redundancy measurements is largely unprecedented. The approach presented in this paper uses a straightforward program synthesis technique: a *compression plan*, consisting of instructions for each block of input data, is generated, guided by the statistical properties of the input data. Because of its use of algorithms specifically suited to the types of redundancy exhibited by the particular input file, the system achieves consistent average performance throughout the file, as shown by experimental evidence.

As an example of the type of savings our system produces, consider compressing a heterogeneous file (such as a small multimedia data file) consisting of 10K of low redundancy (non-natural language) ASCII data, 10K of English text, and 25K of graphics. In this case, a reasonably sophisticated compression program might recognize the increased savings achievable by employing Huffman compression, to better take advantage of the fact that the majority of the data is graphical. However, none of the general-purpose compression methods under consideration are optimal when used alone on this file. This is because the text part of this file is best compressed by textual substitution methods (e.g., Lempel–Ziv) rather than statistical methods, while the low-redundancy data* and graphics parts are best compressed by alphabetic distribution-based methods (e.g., arithmetic or dynamic Huffman coding) rather than Lempel–Ziv or run-length encoding. This particular file totals 45K in length before compression. A compressor using pure dynamic Huffman coding only achieves about 7 per cent savings for a compressed file of length 42.2K. One of the best general-purpose Lempel–Ziv compressors currently available[8,9] achieves 18 per cent savings, producing a compressed file of length 37.4K. Our system uses arithmetic coding on the first and last segments and Lempel–Ziv compression on the text segment in the middle, achieving a 22 per cent savings and producing a compressed file of length 35.6K. This is a 4 per cent improvement over the best commercial system.

* This denotes, in our system, a file with a low rate of repeated strings.

DOCKET ALARM

Explore Litigation Insights

Docket Alarm provides insights to develop a more informed litigation strategy and the peace of mind of knowing you're on top of things.

Real-Time Litigation Alerts

Keep your litigation team up-to-date with **real-time alerts** and advanced team management tools built for the enterprise, all while greatly reducing PACER spend.

Our comprehensive service means we can handle Federal, State, and Administrative courts across the country.

Advanced Docket Research

With over 230 million records, Docket Alarm's cloud-native docket research platform finds what other services can't. Coverage includes Federal, State, plus PTAB, TTAB, ITC and NLRB decisions, all in one place.

Identify arguments that have been successful in the past with full text, pinpoint searching. Link to case law cited within any court document via Fastcase.

Analytics At Your Fingertips

Learn what happened the last time a particular judge, opposing counsel or company faced cases similar to yours.

Advanced out-of-the-box PTAB and TTAB analytics are always at your fingertips.

API

Docket Alarm offers a powerful API (application programming interface) to developers that want to integrate case filings into their apps.

LAW FIRMS

Build custom dashboards for your attorneys and clients with live data direct from the court.

Automate many repetitive legal tasks like conflict checks, document management, and marketing.

FINANCIAL INSTITUTIONS

Litigation and bankruptcy checks for companies and debtors.

E-DISCOVERY AND LEGAL VENDORS

Sync your system to PACER to automate legal marketing.

WHAT WILL YOU BUILD? | **sales@docketalarm.com** | **1-866-77-FASTCASE**

fastcase®
Smarter legal research.