Celso C. Ribeiro
Simone L. Martins (Eds.)

LNCS 3059

Experimental and Efficient Algorithms

Third International Workshop, WEA 2004
Angra dos Reis, Brazil, May 2004
Proceedings

Springer

Celso C. Ribeiro   Simone L. Martins (Eds.)

Experimental and Efficient Algorithms

Third International Workshop, WEA 2004
Angra dos Reis, Brazil, May 25-28, 2004
Proceedings

Springer

Volume Editors

Celso C. Ribeiro
Simone L. Martins
Universidade Federal Fluminense
Department of Computer Science
Niterói, RJ 24210-240, Brazil
E-mail: {celso,simone}@ic.uff.br

Library of Congress Control Number: 2004105538

CR Subject Classification (1998): F.2.1-2, E.1, G.1-2, I.3.5, I.2.8

ISSN 0302-9743
ISBN 3-540-22067-4 Springer-Verlag Berlin Heidelberg New York

Springer-Verlag is a part of Springer Science+Business Media

springeronline.com

Printed in Germany

Typesetting: Camera-ready by author, data conversion by PTP-Berlin, Protago-TeX-Production GmbH
Printed on acid-free paper SPIN: 11008408 06/3142 5 4 3 2 1 0

Springer
Berlin
Heidelberg
New York
Hong Kong
London
Milan
Paris
Tokyo

Lecture Notes in Computer Science

For information about Vols. 1–2951

please contact your bookseller or Springer-Verlag

Vol. 3083: W. Emmerich, A.L. Wolf (Eds.), Component Deployment. X, 249 pages. 2004.

Vol. 3063: A. Llamosí, A. Strohmeier (Eds.), Reliable Software Technologies - Ada-Europe 2004. XIII, 333 pages. 2004.

Vol. 3060: A.Y. Tawfik, S.D. Goodwin (Eds.), Advances in Artificial Intelligence. XIII, 582 pages. 2004. (Subseries LNAI).

Vol. 3059: C.C. Ribeiro, S.L. Martins (Eds.), Experimental and Efficient Algorithms. X, 586 pages. 2004.

Vol. 3058: N. Sebe, M.S. Lew, T.S. Huang (Eds.), Computer Vision in Human-Computer Interaction. X, 233 pages. 2004.

Vol. 3056: H. Dai, R. Srikant, C. Zhang (Eds.), Advances in Knowledge Discovery and Data Mining. XIX, 713 pages. 2004. (Subseries LNAI).

Vol. 3054: I. Crnkovic, J.A. Stafford, H.W. Schmidt, K. Wallnau (Eds.), Component-Based Software Engineering. XI, 311 pages. 2004.

Vol. 3053: C. Bussler, J. Davies, D. Fensel, R. Studer (Eds.), The Semantic Web: Research and Applications. XIII, 490 pages. 2004.

Vol. 3046: A. Laganà (Ed.), Computational Science and Its Applications - ICCSA 2004. LIII, 1016 pages. 2004.

Vol. 3045: A. Laganà, M.L. Gavrilova, V. Kumar, Y. Mun, C.K. Tan, O. Gervasi (Eds.), Computational Science and Its Applications – ICCSA 2004. LIII, 1040 pages. 2004.

Vol. 3044: A. Laganà, M.L. Gavrilova, V. Kumar, Y. Mun, C.K. Tan, O. Gervasi (Eds.), Computational Science and Its Applications – ICCSA 2004. LIII, 1140 pages. 2004.

Vol. 3043: A. Laganà, M.L. Gavrilova, V. Kumar, Y. Mun, C.K. Tan, O. Gervasi (Eds.), Computational Science and Its Applications – ICCSA 2004. LIII, 1180 pages. 2004.

Vol. 3042: N. Mitrou, K. Kontovasilis, G.N. Rouskas, I. Iliadis, L. Merakos (Eds.), NETWORKING 2004, Networking Technologies, Services, and Protocols; Performance of Computer and Communication Networks; Mobile and Wireless Communications. XXXIII, 1519 pages. 2004.

Vol. 3035: M.A. Wimmer, Knowledge Management in Electronic Government. XII, 342 pages. 2004. (Subseries LNAI).

Vol. 3034: J. Favela, E. Menasalvas, E. Chávez (Eds.), Advances in Web Intelligence. XIII, 227 pages. 2004. (Subseries LNAI).

Vol. 3033: M. Li, X.-H. Sun, Q. Deng, J. Ni (Eds.), Grid and Cooperative Computing. XXXVIII, 1076 pages. 2004.

Vol. 3032: M. Li, X.-H. Sun, Q. Deng, J. Ni (Eds.), Grid and Cooperative Computing. XXXVII, 1112 pages. 2004.

Vol. 3031: A. Butz, A. Krüger, P. Olivier (Eds.), Smart Graphics. X, 165 pages. 2004.

Vol. 3028: D. Neuenschwander, Probabilistic and Statistical Methods in Cryptology. X, 158 pages. 2004.

Vol. 3027: C. Cachin, J. Camenisch (Eds.), Advances in Cryptology - EUROCRYPT 2004. XI, 628 pages. 2004.

Vol. 3026: C. Ramamoorthy, R. Lee, K.W. Lee (Eds.), Software Engineering Research and Applications. XV, 377 pages. 2004.

Vol. 3025: G.A. Vouros, T. Panayiotopoulos (Eds.), Methods and Applications of Artificial Intelligence. XV, 546 pages. 2004. (Subseries LNAI).

Vol. 3024: T. Pajdla, J. Matas (Eds.), Computer Vision - ECCV 2004. XXVIII, 621 pages. 2004.

Vol. 3023: T. Pajdla, J. Matas (Eds.), Computer Vision - ECCV 2004. XXVIII, 611 pages. 2004.

Vol. 3022: T. Pajdla, J. Matas (Eds.), Computer Vision - ECCV 2004. XXVIII, 621 pages. 2004.

Vol. 3021: T. Pajdla, J. Matas (Eds.), Computer Vision - ECCV 2004. XXVIII, 633 pages. 2004.

Vol. 3019: R. Wyrzykowski, J. Dongarra, M. Paprzycki, J. Wasniewski (Eds.), Parallel Processing and Applied Mathematics. XIX, 1174 pages. 2004.

Vol. 3015: C. Barakat, I. Pratt (Eds.), Passive and Active Network Measurement. XI, 300 pages. 2004.

Vol. 3012: K. Kurumatani, S.-H. Chen, A. Ohuchi (Eds.), Multi-Agnets for Mass User Support. X, 217 pages. 2004. (Subseries LNAI).

Vol. 3011: J.-C. Régin, M. Rueher (Eds.), Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems. XI, 415 pages. 2004.

Vol. 3010: K.R. Apt, F. Fages, F. Rossi, P. Szeredi, J. Váncza (Eds.), Recent Advances in Constraints. VIII, 285 pages. 2004. (Subseries LNAI).

Vol. 3009: F. Bomarius, H. Iida (Eds.), Product Focused Software Process Improvement. XIV, 584 pages. 2004.

Vol. 3008: S. Heuel, Uncertain Projective Geometry. XVII, 205 pages. 2004.

Vol. 3007: J.X. Yu, X. Lin, H. Lu, Y. Zhang (Eds.), Advanced Web Technologies and Applications. XXII, 936 pages. 2004.

Vol. 3006: M. Matsui, R. Zuccherato (Eds.), Selected Areas in Cryptography. XI, 361 pages. 2004.

Vol. 3005: G.R. Raidl, S. Cagnoni, J. Branke, D.W. Corne, R. Drechsler, Y. Jin, C.G. Johnson, P. Machado, E. Marchiori, F. Rothlauf, G.D. Smith, G. Squillero (Eds.), Applications of Evolutionary Computing. XVII, 562 pages. 2004.

Vol. 3004: J. Gottlieb, G.R. Raidl (Eds.), Evolutionary Computation in Combinatorial Optimization. X, 241 pages. 2004.

Vol. 3003: M. Keijzer, U.-M. O'Reilly, S.M. Lucas, E. Costa, T. Soule (Eds.), Genetic Programming. XI, 410 pages. 2004.

Vol. 3002: D.L. Hicks (Ed.), Metainformatics. X, 213 pages. 2004.

Vol. 3001: A. Ferscha, F. Mattern (Eds.), Pervasive Computing. XVII, 358 pages. 2004.

Vol. 2999: E.A. Boiten, J. Derrick, G. Smith (Eds.), Integrated Formal Methods. XI, 541 pages. 2004.

Vol. 2998: Y. Kameyama, P.J. Stuckey (Eds.), Functional and Logic Programming. X, 307 pages. 2004.

Vol. 2997: S. McDonald, J. Tait (Eds.), Advances in Information Retrieval. XIII, 427 pages. 2004.

Vol. 2996: V. Diekert, M. Habib (Eds.), STACS 2004. XVI, 658 pages. 2004.

Vol. 2995: C. Jensen, S. Poslad, T. Dimitrakos (Eds.), Trust Management. XIII, 377 pages. 2004.

Vol. 2994: E. Rahm (Ed.), Data Integration in the Life Sciences. X, 221 pages. 2004. (Subseries LNBI).

Vol. 2993: R. Alur, G.J. Pappas (Eds.), Hybrid Systems: Computation and Control. XII, 674 pages. 2004.

Vol. 2992: E. Bertino, S. Christodoulakis, D. Plexousakis, V. Christophides, M. Koubarakis, K. Böhm, E. Ferrari (Eds.), Advances in Database Technology - EDBT 2004. XVIII, 877 pages. 2004.

Vol. 2991: R. Alt, A. Frommer, R.B. Kearfott, W. Luther (Eds.), Numerical Software with Result Verification. X, 315 pages. 2004.

Vol. 2989: S. Graf, L. Mounier (Eds.), Model Checking Software. X, 309 pages. 2004.

Vol. 2988: K. Jensen, A. Podelski (Eds.), Tools and Algorithms for the Construction and Analysis of Systems. XIV, 608 pages. 2004.

Vol. 2987: I. Walukiewicz (Ed.), Foundations of Software Science and Computation Structures. XIII, 529 pages. 2004.

Vol. 2986: D. Schmidt (Ed.), Programming Languages and Systems. XII, 417 pages. 2004.

Vol. 2985: E. Duesterwald (Ed.), Compiler Construction. X, 313 pages. 2004.

Vol. 2984: M. Wermelinger, T. Margaria-Steffen (Eds.), Fundamental Approaches to Software Engineering. XII, 389 pages. 2004.

Vol. 2983: S. Istrail, M.S. Waterman, A. Clark (Eds.), Computational Methods for SNPs and Haplotype Inference. IX, 153 pages. 2004. (Subseries LNBI).

Vol. 2982: N. Wakamiya, M. Solarski, J. Sterbenz (Eds.), Active Networks. XI, 308 pages. 2004.

Vol. 2981: C. Müller-Schloer, T. Ungerer, B. Bauer (Eds.), Organic and Pervasive Computing – ARCS 2004. XI, 339 pages. 2004.

Vol. 2980: A. Blackwell, K. Marriott, A. Shimojima (Eds.), Diagrammatic Representation and Inference. XV, 448 pages. 2004. (Subseries LNAI).

Vol. 2979: I. Stoica, Stateless Core: A Scalable Approach for Quality of Service in the Internet. XVI, 219 pages. 2004.

Vol. 2978: R. Groz, R.M. Hierons (Eds.), Testing of Communicating Systems. XII, 225 pages. 2004.

Vol. 2977: G. Di Marzo Serugendo, A. Karageorgos, O.F. Rana, F. Zambonelli (Eds.), Engineering Self-Organising Systems. X, 299 pages. 2004. (Subseries LNAI).

Vol. 2976: M. Farach-Colton (Ed.), LATIN 2004: Theoretical Informatics. XV, 626 pages. 2004.

Vol. 2973: Y. Lee, J. Li, K.-Y. Whang, D. Lee (Eds.), Database Systems for Advanced Applications. XXIV, 925 pages. 2004.

Vol. 2972: R. Monroy, G. Arroyo-Figueroa, L.E. Sucar, H. Sossa (Eds.), MICAI 2004: Advances in Artificial Intelligence. XVII, 923 pages. 2004. (Subseries LNAI).

Vol. 2971: J.I. Lim, D.H. Lee (Eds.), Information Security and Cryptology -ICISC 2003. XI, 458 pages. 2004.

Vol. 2970: F. Fernández Rivera, M. Bubak, A. Gómez Tato, R. Doallo (Eds.), Grid Computing. XI, 328 pages. 2004.

Vol. 2968: J. Chen, S. Hong (Eds.), Real-Time and Embedded Computing Systems and Applications. XIV, 620 pages. 2004.

Vol. 2967: S. Melnik, Generic Model Management. XX, 238 pages. 2004.

Vol. 2966: F.B. Sachse, Computational Cardiology. XVIII, 322 pages. 2004.

Vol. 2965: M.C. Calzarossa, E. Gelenbe, Performance Tools and Applications to Networked Systems. VIII, 385 pages. 2004.

Vol. 2964: T. Okamoto (Ed.), Topics in Cryptology – CT-RSA 2004. XI, 387 pages. 2004.

Vol. 2963: R. Sharp, Higher Level Hardware Synthesis. XVI, 195 pages. 2004.

Vol. 2962: S. Bistarelli, Semirings for Soft Constraint Solving and Programming. XII, 279 pages. 2004.

Vol. 2961: P. Eklund (Ed.), Concept Lattices. IX, 411 pages. 2004. (Subseries LNAI).

Vol. 2960: P.D. Mosses (Ed.), CASL Reference Manual. XVII, 528 pages. 2004.

Vol. 2959: R. Kazman, D. Port (Eds.), COTS-Based Software Systems. XIV, 219 pages. 2004.

Vol. 2958: L. Rauchwerger (Ed.), Languages and Compilers for Parallel Computing. XI, 556 pages. 2004.

Vol. 2957: P. Langendoerfer, M. Liu, I. Matta, V. Tsaoussidis (Eds.), Wired/Wireless Internet Communications. XI, 307 pages. 2004.

Vol. 2956: A. Dengel, M. Junker, A. Weisbecker (Eds.), Reading and Learning. XII, 355 pages. 2004.

Vol. 2954: F. Crestani, M. Dunlop, S. Mizzaro (Eds.), Mobile and Ubiquitous Information Access. X, 299 pages. 2004.

Vol. 2953: K. Konrad, Model Generation for Natural Language Interpretation and Analysis. XIII, 166 pages. 2004. (Subseries LNAI).

Vol. 2952: N. Guelfi, E. Astesiano, G. Reggio (Eds.), Scientific Engineering of Distributed Java Applications. X, 157 pages. 2004.

Preface

The Third International Workshop on Experimental and Efficient Algorithms (WEA 2004) was held in Angra dos Reis (Brazil), May 25–28, 2004.

The WEA workshops are sponsored by the European Association for Theoretical Computer Science (EATCS). They are intended to provide an international forum for researchers in the areas of design, analysis, and experimental evaluation of algorithms. The two preceding workshops in this series were held in Riga (Latvia, 2001) and Ascona (Switzerland, 2003).

This proceedings volume comprises 40 contributed papers selected by the Program Committee along with the extended abstracts of the invited lectures presented by Richard Karp (University of California at Berkeley, USA), Giuseppe Italiano (University of Rome "Tor Vergata", Italy), and Christos Kaklamanis (University of Patras, Greece).

As the organizer and chair of this wokshop, I would like to thank all the authors who generously supported this project by submitting their papers for publication in this volume. I am also grateful to the invited lecturers, who kindly accepted our invitation.

For their dedication and collaboration in the refereeing procedure, I would like also to express my gratitude to the members of the Program Committee: E. Amaldi (Italy), J. Blazewicz (Poland), V.-D. Cung (France), U. Derigs (Germany), J. Diaz (Spain), M. Gendreau (Canada), A. Goldberg (USA), P. Hansen (Canada), T. Ibaraki (Japan), K. Jansen (Germany), S. Martello (Italy), C.C. McGeoch (USA), L.S. Ochi (Brazil), M.G.C. Resende (USA), J. Rolim (Switzerland), S. Skiena (USA), M. Sniedovich (Australia), C.C. Souza (Brazil), P. Spirakis (Greece), D. Trystram (France), and S. Voss (Germany). I am also grateful to the anonymous referees who assisted the Program Committee in the selection of the papers to be included in this publication.

The idea of organizing WEA 2004 in Brazil grew out of a few meetings with José Rolim (University of Geneva, Switzerland). His encouragement and close collaboration at different stages of this project were fundamental for the success of the workshop. The support of EATCS and Alfred Hofmann (Springer-Verlag) were also appreciated.

I am thankful to the Department of Computer Science of *Universidade Federal Fluminense* (Niterói, Brazil) for fostering the environment in which this workshop was organized. I am particularly indebted to Simone Martins for her invaluable support and collaboration in the editorial work involved in the preparation of the final camera-ready copy of this volume.

Angra dos Reis (Brazil), May 2004 Celso C. Ribeiro (Chair)

Table of Contents

A Hybrid Bin-Packing Heuristic to Multiprocessor Scheduling

Adriana C.F. Alvim[1] and Celso C. Ribeiro[2]

[1] Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia de Produção, PO Box 68507, Rio de Janeiro 21945-970, Brazil.
alvim@inf.puc-rio.br

[2] Universidade Federal Fluminense, Department of Computer Science, Rua Passo da Pátria 156, Niterói, RJ 24210-240, Brazil.
celso@inf.puc-rio.br

Abstract. The multiprocessor scheduling problem consists in scheduling a set of tasks with known processing times into a set of identical processors so as to minimize their makespan, i.e., the maximum processing time over all processors. We propose a new heuristic for solving the multiprocessor scheduling problem, based on a hybrid heuristic to the bin packing problem. Computational results illustrating the effectiveness of this approach are reported and compared with those obtained by other heuristics.

1 Introduction

Let $T = \{T_1, \dots, T_n\}$ be a set of n tasks with processing times t_i, $i = 1, \dots, n$, to be processed by a set $P = \{P_1, \dots, P_m\}$ of $m \geq 2$ identical processors. We assume the processing times are nonnegative integers satisfying $t_1 \geq t_2 \geq \dots \geq t_n$. Each processor can handle at most one task at any given time and preemption is not possible. We denote by A_j the set formed by the indices of the tasks assigned to processor P_j and by $t(P_j) = \sum_{i \in A_j} t_i$ its overall processing time, $j = 1, \dots, n$. A solution is represented by the lists of tasks assigned to each processor. The makespan of a solution $S = (A_1, \dots, A_m)$ is given by $C_{\max}(S) = \max_{j=1,\dots,m} t(P_j)$.

The multiprocessor scheduling problem $P||C_{\max}$ consists in finding an optimal assignment of tasks to processors, so as to minimize their makespan, see e.g. [5,16,20]. $P||C_{\max}$ is NP-hard [4,14]. We denote the optimal makespan by $C^*_{\max}$. Minimizing the schedule length is important since it leads to the maximization of the processor utilization factor [3].

There is a duality relation [5,17,18,24] between $P||C_{\max}$ and the bin packing problem (BP), which consists in finding the minimum number of bins of a given capacity C which are necessary to accommodate n items with weights $t_1, \dots, t_n$.

The worst case performance ratio $r(\mathrm{H})$ of a given heuristic H for $P||C_{\max}$ is defined as the maximum value of the ratio $\mathrm{H}(I)/C^*_{\max}(I)$ over all instances I, where $C^*_{\max}(I)$ is the optimal makespan of instance I and $\mathrm{H}(I)$ is the makespan of the solution computed by heuristic H. The longest processing time (LPT)

C.C. Ribeiro and S.L. Martins (Eds.): WEA 2004, LNCS 3059, pp. 1–13, 2004.

heuristic of Graham [15] finds an approximate solution to $P||C_{\max}$ in time $O(n \log n + n \log m)$, with $r(\text{LPT}) = 4/3 - 1/3m$. The MULTIFIT heuristic proposed by Coffman et al. [6] explores the duality relation with the bin packing problem, searching by binary search the minimum processing time (or bin capacity) such that the solution obtained by the FFD heuristic [19,21] to pack the n tasks (or items) makes use of at most m processors (or bins). It can be shown that if MULTIFIT is applied k times, then it runs in time $O(n \log n + kn \log m)$ and $r(\text{MULTIFIT}) = 1.22 + 2^{-k}$. Friesen [13] subsequently improved this ratio to $1.20 + 2^{-k}$. Yue [25] further improved it to 13/11, which is tight. Finn and Horowitz [9] proposed the 0/1-INTERCHANGE heuristic running in time $O(n \log m)$, with worst case performance ratio equal to 2. Variants and extensions of the above heuristics can be found in the literature.

The duality relation between the bin packing problem and $P||C_{\max}$ was also used by Alvim and Ribeiro [1] to derive a hybrid improvement heuristic to the former. This heuristic is explored in this paper in the context of $P||C_{\max}$. Lower and upper bounds used by the heuristic are described in Section 2. The main steps of the hybrid improvement heuristic to multiprocessor scheduling are presented in Section 3. Numerical results illustrating the effectiveness of the proposed algorithm are reported in Section 4. Concluding remarks are made in the last section.

2 Lower and Upper Bounds

The lower bound $L_1 = \max\left\{\left\lceil \frac{1}{m}\sum_{i=1}^{n} p_i \right\rceil, \max_{i=1,\dots,n}\{p_i\}\right\}$ proposed by McNaughton [22] establishes that the optimal makespan cannot be smaller than the maximum between the average processing time over all processors and the longest duration over all tasks. This bound can be further improved to $L_2 = \max\left\{L_1,\ p_m + p_{m+1}\right\}$.

Dell'Amico and Martello [7] proposed the lower bound L_3. They also showed that the combination of lower and upper bounds to the makespan makes it possible to derive lower and upper bounds to the number of tasks assigned to each processor, leading to a new lower bound L_ϑ. Bounds L_2, L_3, and L_ϑ are used in the heuristic described in the next section.

We used three construction procedures for building feasible solutions and computing upper bounds to $P||C_{\max}$:

- Construction heuristic H1: Originally proposed in [1,2], it is similar to the Multi-Subset heuristic in [7]. It considers one processor at a time. The longest yet unassigned task is assigned to the current processor. Next, assign to this same processor a subset of the yet unassigned tasks such that the sum of their processing times is as close as possible to a given limit to the makespan. The polynomial-time approximation scheme MTSS(3) of Martello and Toth [21] is used in this step. The remaining unassigned tasks are considered one by one in non-increasing order of their processing times. Each of them is assigned to the processor with the smallest load.

- Construction heuristic H2: Hochbaum and Shmoys [17] proposed a new approach to constructing approximation algorithms, called dual approximation algorithms. The goal is to find superoptimal, but infeasible, solutions. They showed that finding an ϵ-approximation to $P||C_{\max}$ is equivalent to finding an ϵ-dual-approximation to BP. For the latter, an ϵ-dual-approximation algorithm constructs a solution in which the number of bins is at most the optimal number of bins and each bin is filled with at most $1+\epsilon$ (bin capacity $C = 1$ and item weights t_i scaled by t_i/C). In particular, they proposed a scheme for $\epsilon = 1/5$. Given a lower bound L and an upper bound U to $C^*_{\max}$, we obtain by binary search the smallest value C such that $L \leq C \leq U$ and the 1/5-dual-approximation solution to BP uses no more than m bins. This approach characterizes a $1/5 + 2^{-k}$-approximation algorithm to $P||C_{\max}$, where k is the number of iterations of the search. At the end of the search, the value of C gives the L_{HS} lower bound to the makespan.
- Construction heuristic H3: This is the longest processing time heuristic LPT [15]. Tasks are ordered in non-increasing order of their processing times. Each task is assigned to the processor with the smallest total processing time.

França et al. [10] proposed algorithm 3-PHASE based on the idea of balancing the load between pair of processors. Hübscher and Glover [18] also explored the relation between $P||C_{\max}$ and BP, proposing a tabu search algorithm using 2-exchanges and influential diversification. Dell'Amico and Martello [7] developed a branch-and-bound algorithm to exactly solve $P||C_{\max}$. They also obtained new lower bounds. Scholl and Voss [24] considered two versions of the simple assembly line balancing problem. If the precedence constraints are not taken into account, these two versions correspond to BP and $P||C_{\max}$. Fatemi-Ghomi and Jolai-Ghazvini [8] proposed a local search algorithm using a neighborhood defined by exchanges of pairs of tasks in different processors. Frangioni et al. [12] proposed new local search algorithms for the minimum makespan processor scheduling problem, which perform multiple exchanges of jobs among machines. The latter are modelled as special disjoint cycles and paths in a suitably defined improvement graph. Several algorithms for searching the neighborhood are suggested and computational experiments are performed for the case of identical processors.

3 Hybrid Improvement Heuristic to $P||C_{\max}$

The hybrid improvement heuristic to $P||C_{\max}$ is described in this section. The core of this procedure is formed by the construction, redistribution, and improvement phases, as illustrated by the pseudo-code of procedure `C+R+I` in Figure 1. It depends on two parameters: the target makespan `Target` and the maximum number of iterations `MaxIterations` performed by the tabu search procedure used in the improvement phase. The loop in lines 1–8 makes three attempts to build a feasible solution S to the bin packing problem defined by the processing times $t_i, i = 1, \ldots, n$, with bin capacity `Target`, using exactly m bins. Each of the heuristics H1, H2, and H3 is used at each attempt in line 2. If S is feasible to the associated bin packing problem, then it is returned in line 3. Otherwise, load

```
procedure C+R+I(Target,MaxIterations);
1 for k = 1,2,3 do
2     Build a solution S = {A_1,...,A_m} to P||C_max using heuristic Hk;
3     if C_max(S) ≤ Target then return S;
4     S ← Redistribution(S);
5     if C_max(S) ≤ Target then return S;
6     S ← TabuSearch(S,MaxIterations);
7     if C_max(S) ≤ Target then return S;
8 end
9 return S;
end C+R+I
```

Fig. 1. Pseudo-code of the core C+R+I procedure.

redistribution is performed in line 4 to improve processor usability and the modified solution S is returned in line 5 if it is feasible to the bin packing problem. Finally, a tabu search procedure is applied in line 6 as an attempt to knock down the makespan of the current solution and the modified solution S is returned in line 7 if it is feasible to the bin packing problem. Detailed descriptions of the redistribution and improvement phases are reported in [1].

The pseudo-code of the complete hybrid improvement heuristic HI_PCmax to $P||C_{\max}$ is given in Figure 2. An initial solution S is built in line 1 using heuristic H3. The lower bound L_2 is computed in line 2. If the current lower and upper bounds coincide, then solution S is returned in line 3. The lower bound L_3 is computed in line 4 and the current lower bound is updated. If the current lower and upper bounds coincide, then solution S is returned in line 5. The lower bound L_ϑ is computed in line 6 and the current lower bound is updated. If the current lower and upper bounds coincide, then solution S is returned in line 7. A new solution S' is built in line 8 using heuristic H2. The currently best solution and the current upper bound are updated in line 9, while the current lower bound is updated in line 10. If the current lower and upper bounds coincide, then the currently best solution S is returned in line 11. A new solution S' is built in line 12 using heuristic H1. The currently best solution and the current upper bound are updated in line 13. If the current lower and upper bounds coincide, then the currently best solution S is returned in line 14. At this point, UB is the upper bound associated with the currently best known solution S to $P||C_{\max}$ and LB is an unattained makespan. The core procedure C+R+I makes an attempt to build a solution with makespan equal to the current lower bound in line 15. The currently best solution and the current upper bound are updated in line 16. If the current lower and upper bounds coincide, then the currently best solution S is returned in line 17. The loop in lines 18–23 implements a binary search strategy seeking for progressively better solutions. The target makespan $C_{\max} = \lfloor (LB + UB)/2 \rfloor$ is set in line 19. Let S' be the solution obtained by the core procedure C+R+I applied in line 20 using $C_{\max}$ as the target makespan. If its makespan is at least as good as the target makespan

$C_{\max}$, then the current upper bound UB and the currently best solution S are updated in line 21. Otherwise, the unattained makespan LB is updated in line 22, since the core procedure C+R+I was not able to find a feasible solution with the target makespan. The best solution found S is returned in line 24.

The core procedure C+R+I is applied at two different points: once in line 15 using the lower bound LB as the target makespan and in line 20 at each iteration of the binary search strategy using $C_{\max}$ as the target makespan. This implementation follows the same EBS (binary search with prespecified entry point) scheme suggested in [24]. Computational experiments have shown that it is able to find better solutions in smaller computation times than other variants which do not explore the binary search strategy or do not make a preliminary attempt to build a solution using LB as the target makespan.

4 Computational Experiments

All computational experiments were performed on a 2.4 GHz AMD XP machine with 512 MB of RAM memory.

procedure HI_PCmax(MaxIterations);
1 Compute a solution S using heuristic H3 and set $UB \leftarrow C_{\max}(S)$;
2 Compute the lower bound L_2 and set $LB \leftarrow L_2$;
3 **if** $LB = UB$ **then return** S;
4 Compute L_3 using binary search in the interval $[LB, UB]$ and set $LB \leftarrow L_3$;
5 **if** $LB = UB$ **then return** S;
6 Compute L_ϑ using LB and UB and update $LB \leftarrow \max\{LB, L_\vartheta\}$;
7 **if** $LB = UB$ **then return** S;
8 Compute a solution S' and the lower bound L_{HS} using heuristic H2;
9 **if** $C_{\max}(S') < UB$ **then** set $UB \leftarrow C_{\max}(S')$ and $S \leftarrow S'$;
10 Update $LB \leftarrow \max\{LB, L_{HS}\}$;
11 **if** $LB = UB$ **then return** S
12 Compute a solution S' using heuristic H1;
13 **if** $C_{\max}(S') < UB$ **then** set $UB \leftarrow C_{\max}(S')$ and $S \leftarrow S'$;
14 **if** $LB = UB$ **then return** S;
15 $S' \leftarrow$ C+R+I(LB, MaxIterations);
16 **if** $C_{\max}(S') < UB$ **then** set $UB \leftarrow C_{\max}(S')$ and $S \leftarrow S'$;
17 **if** $LB = UB$ **then return** S;
18 **while** $LB < UB - 1$ **do**
19     $C_{\max} \leftarrow \lfloor (LB + UB)/2 \rfloor$;
20     $S' \leftarrow$ C+R+I($C_{\max}$, MaxIterations);
21   **if** $C_{\max}(S') \leq C_{\max}$ **then** set $UB \leftarrow C_{\max}(S')$ and $S \leftarrow S'$;
22   **else** $LB \leftarrow C_{\max}$;
23 **end**
24 **return** S;
end HI_PCmax

Fig. 2. Pseudo-code of the hybrid improvement procedure to $P||C_{\max}$.