

Siu-Wing Cheng
Chung Keung Poon (Eds.)

LNCS 4041

Algorithmic Aspects in Information and Management

Second International Conference, AAIM 2006
Hong Kong, China, June 2006
Proceedings



Springer

TP301.6-53
A111
2006

Siu-Wing Cheng Chung Keung Poon (Eds.)

Algorithmic Aspects in Information and Management

Second International Conference, AAIM 2006
Hong Kong, China, June 20-22, 2006
Proceedings



Springer



E200603640

Volume Editors

Siu-Wing Cheng

Hong Kong University of Science and Technology, Department of Computer Science
Clear Water Bay, Hong Kong, China
E-mail: scheng@cs.ust.hk

Chung Keung Poon

City University of Hong Kong, Department of Computer Science
83 Tat Chee Avenue, Kowloon Tong, Hong Kong, China
E-mail: ckpoon@cs.cityu.edu.hk

Library of Congress Control Number: 2006927039

CR Subject Classification (1998): F.2.1-2, E.1, G.1-3, J.1

LNCS Sublibrary: SL 3 – Information Systems and Application, incl. Internet/Web and HCI

ISSN 0302-9743

ISBN-10 3-540-35157-4 Springer Berlin Heidelberg New York

ISBN-13 978-3-540-35157-3 Springer Berlin Heidelberg New York

This work is subject to copyright. All rights are reserved, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, re-use of illustrations, recitation, broadcasting, reproduction on microfilms or in any other way, and storage in data banks. Duplication of this publication or parts thereof is permitted only under the provisions of the German Copyright Law of September 9, 1965, in its current version, and permission for use must always be obtained from Springer. Violations are liable to prosecution under the German Copyright Law.

Springer is a part of Springer Science+Business Media

springer.com

© Springer-Verlag Berlin Heidelberg 2006

Printed in Germany

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper SPIN: 11775096 06/3142 5 4 3 2 1 0

Preface

The papers contained in this volume were presented at the Second International Conference on Algorithmic Aspects in Information and Management (AAIM 2006), held on June 20–22, 2006 at the City University of Hong Kong, Hong Kong, China.

The series of AAIM conferences provides an annual international forum for the communication of research advances on algorithms pertinent to information management and management science. The first conference (AAIM 2005) was held in Xi'an, China and it is planned for the near future that conferences of the series will be held in cities in the Pacific Rim.

This volume contains 34 papers selected from a total of 263 papers submitted from places all over the world: Australia, Canada, China, France, Germany, India, Israel, Italy, Japan, Mexico, Mongolia, Netherlands, New Zealand, Poland, Singapore, South Korea, Sweden, Taiwan, Ukraine, UK and USA. In addition to the selected papers, the volume also contains two papers by the invited speakers, Allan Borodin and Ming-Yang Kao.

We thank all the people who made this meeting possible: the authors who submitted papers, the Program Committee members and external reviewers, the invited speakers, the local organizers, and the sponsors for their effort, advice and support. We also thank EasyChair (www.easychair.org) for providing the free conference software.

April 2006

Siu-Wing Cheng
Chung Keung Poon

Conference Organization

AAIM 2006 was jointly organized by the City University of Hong Kong and the Hong Kong University of Science and Technology.

Program Chairs

Siu-Wing Cheng (Hong Kong U of Science and Technology)
Chung Keung Poon (City U of Hong Kong)

Program Committee

Hee-Kap Ahn (Korean Advanced Institute of Science and Technology)
Takao Asano (Chuo U)
Amotz Bar-Noy (City U of New York)
Hans Bodlaender (U of Utrecht)
Peter Brucker (U of Osnabrueck)
Leizhen Cai (Chinese U of Hong Kong)
Jianer Chen (Texas A&M U)
Marek Chrobak (U of California at Riverside)
Rudolf Fleischer (Fudan U)
Joachim Gudmundsson (National ICT Australia)
Gregory Gutin (Royal Holloway, U of London and U of Haifa)
Wen-Lian Hsu (Academia Sinica, Taiwan)
Giuseppe F. Italiano (U of Rome “Tor Vergata”)
Tao Jiang (U of California at Riverside and Tsinghua U)
Tak-Wah Lam (U of Hong Kong)
Xiang-Yang Li (Illinois Institute of Technology)
Peter Bro Miltersen (U of Aarhus)
Pat Morin (Carleton U)
Seffi Naor (Technion and Microsoft Research)
Kirk Pruhs (U of Pittsburgh)
Vijaya Ramachandran (U of Texas at Austin)
Rajeev Raman (Leicester U)
Jiri Sgall (Academy of Sciences of Czech Republic)
Paul Spirakis (U of Patras and CTI Greece)
Wing Kin Sung (National U of Singapore)
Hisao Tamaki (Meiji U)
Jan van Leeuwen (U of Utrecht)
Lusheng Wang (City U of Hong Kong)
Yinfeng Xu (Xi'an Jiaotong U)
Binhai Zhu (Montana State U)

External Reviewers

Sang Won Bae
Rezaul Alam Chowdhury
Xiaotie Deng
Andrew Goldberg
Wen-Liang Hwang
Hyunwoo Jung
Yue-Kuen Kwok
James Kwok
Eric Law
Peter Lennartz
Chi-Jen Lu
Esther Moet
Mart Molle
Andrea Pacifici
Guido Proietti
Chan-Su Shin
Maurizio A. Strangio

Organizing Committee

Matthew Chang (City U of Hong Kong)
Mordecai Golin (Hong Kong U of Science and Technology)
Jinxin Huang (Hong Kong U of Science and Technology)
Xiaohua Jia (City U of Hong Kong)
Hing Fung Ting (U of Hong Kong)
Yajun Wang (Hong Kong U of Science and Technology)

Sponsors

City University of Hong Kong
Hong Kong Pei Hua Education Foundation Limited

Commenced Publication in 1973

Founding and Former Series Editors:

Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

Editorial Board

David Hutchison

Lancaster University, UK

Takeo Kanade

Carnegie Mellon University, Pittsburgh, PA, USA

Josef Kittler

University of Surrey, Guildford, UK

Jon M. Kleinberg

Cornell University, Ithaca, NY, USA

Friedemann Mattern

ETH Zurich, Switzerland

John C. Mitchell

Stanford University, CA, USA

Moni Naor

Weizmann Institute of Science, Rehovot, Israel

Oscar Nierstrasz

University of Bern, Switzerland

C. Pandu Rangan

Indian Institute of Technology, Madras, India

Bernhard Steffen

University of Dortmund, Germany

Madhu Sudan

Massachusetts Institute of Technology, MA, USA

Demetri Terzopoulos

University of California, Los Angeles, CA, USA

Doug Tygar

University of California, Berkeley, CA, USA

Moshe Y. Vardi

Rice University, Houston, TX, USA

Gerhard Weikum

Max-Planck Institute of Computer Science, Saarbruecken, Germany

Lecture Notes in Computer Science

For information about Vols. 1–3925

please contact your bookseller or Springer

Vol. 4044: P. Abrahamsson, M. Marchesi, G. Succi (Eds.), *Extreme Programming and Agile Processes in Software Engineering*. XII, 230 pages. 2006.

Vol. 4041: S.-W. Cheng, C.K. Poon (Eds.), *Algorithmic Aspects in Information and Management*. XI, 395 pages. 2006.

Vol. 4039: M. Morisio (Ed.), *Reuse of Off-the-Shelf Components*. XIII, 444 pages. 2006.

Vol. 4038: P. Ciancarini, H. Wiklicky (Eds.), *Coordination Models and Languages*. VIII, 299 pages. 2006.

Vol. 4037: R. Gorrieri, H. Wehrheim (Eds.), *Formal Methods for Open Object-Based Distributed Systems*. XVII, 474 pages. 2006.

Vol. 4034: J. Münch, M. Vierimaa (Eds.), *Product-Focused Software Process Improvement*. XVII, 474 pages. 2006.

Vol. 4027: H.L. Larsen, G. Pasi, D. Ortiz-Arroyo, T. Andreassen, H. Christiansen (Eds.), *Flexible Query Answering Systems*. XVIII, 714 pages. 2006. (Sublibrary LNAI).

Vol. 4025: F. Eliassen, A. Montresor (Eds.), *Distributed Applications and Interoperable Systems*. XI, 355 pages. 2006.

Vol. 4024: S. Donatelli, P. S. Thiagarajan (Eds.), *Petri Nets and Other Models of Concurrency - ICATPN 2006*. XI, 441 pages. 2006.

Vol. 4021: E. André, L. Dybkjær, W. Minker, H. Neumann, M. Weber (Eds.), *Perception and Interactive Technologies*. XI, 217 pages. 2006. (Sublibrary LNAI).

Vol. 4011: Y. Sure, J. Domingue (Eds.), *The Semantic Web: Research and Applications*. XIX, 726 pages. 2006.

Vol. 4010: S. Dunne, B. Stoddart (Eds.), *Unifying Theories of Programming*. VIII, 257 pages. 2006.

Vol. 4007: C. Álvarez, M. Serna (Eds.), *Experimental Algorithms*. XI, 329 pages. 2006.

Vol. 4006: L.M. Pinho, M. González Harbour (Eds.), *Reliable Software Technologies – Ada-Europe 2006*. XII, 241 pages. 2006.

Vol. 4004: S. Vaudenay (Ed.), *Advances in Cryptology - EUROCRYPT 2006*. XIV, 613 pages. 2006.

Vol. 4003: Y. Koucheryavy, J. Harju, V.B. Iversen (Eds.), *Next Generation Teletraffic and Wired/Wireless Advanced Networking*. XVI, 582 pages. 2006.

Vol. 4001: E. Dubois, K. Pohl (Eds.), *Advanced Information Systems Engineering*. XVI, 560 pages. 2006.

Vol. 3999: C. Kop, G. Fliedl, H.C. Mayr, E. Métais (Eds.), *Natural Language Processing and Information Systems*. XIII, 227 pages. 2006.

Vol. 3998: T. Calamoneri, I. Finocchi, G.F. Italiano (Eds.), *Algorithms and Complexity*. XII, 394 pages. 2006.

Vol. 3997: W. Grieskamp, C. Weise (Eds.), *Formal Approaches to Software Testing*. XII, 219 pages. 2006.

Vol. 3996: A. Keller, J.-P. Martin-Flatin (Eds.), *Self-Managed Networks, Systems, and Services*. X, 185 pages. 2006.

Vol. 3995: G. Müller (Ed.), *Emerging Trends in Information and Communication Security*. XX, 524 pages. 2006.

Vol. 3994: V.N. Alexandrov, G.D. van Albada, P.M.A. Sloot, J. Dongarra (Eds.), *Computational Science – ICCS 2006, Part IV*. XXXV, 1096 pages. 2006.

Vol. 3993: V.N. Alexandrov, G.D. van Albada, P.M.A. Sloot, J. Dongarra (Eds.), *Computational Science – ICCS 2006, Part III*. XXXVI, 1136 pages. 2006.

Vol. 3992: V.N. Alexandrov, G.D. van Albada, P.M.A. Sloot, J. Dongarra (Eds.), *Computational Science – ICCS 2006, Part II*. XXXV, 1122 pages. 2006.

Vol. 3991: V.N. Alexandrov, G.D. van Albada, P.M.A. Sloot, J. Dongarra (Eds.), *Computational Science – ICCS 2006, Part I*. LXXXI, 1096 pages. 2006.

Vol. 3990: J. C. Beck, B.M. Smith (Eds.), *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems*. X, 301 pages. 2006.

Vol. 3989: J. Zhou, M. Yung, F. Bao, *Applied Cryptography and Network Security*. XIV, 488 pages. 2006.

Vol. 3987: M. Hazas, J. Krumm, T. Strang (Eds.), *Location- and Context-Awareness*. X, 289 pages. 2006.

Vol. 3986: K. Stølen, W.H. Winsborough, F. Martinelli, F. Massacci (Eds.), *Trust Management*. XIV, 474 pages. 2006.

Vol. 3984: M. Gavrilova, O. Gervasi, V. Kumar, C.J. K. Tan, D. Taniar, A. Laganà, Y. Mun, H. Choo (Eds.), *Computational Science and Its Applications - ICCSA 2006, Part V*. XXV, 1045 pages. 2006.

Vol. 3983: M. Gavrilova, O. Gervasi, V. Kumar, C.J. K. Tan, D. Taniar, A. Laganà, Y. Mun, H. Choo (Eds.), *Computational Science and Its Applications - ICCSA 2006, Part IV*. XXVI, 1191 pages. 2006.

Vol. 3982: M. Gavrilova, O. Gervasi, V. Kumar, C.J. K. Tan, D. Taniar, A. Laganà, Y. Mun, H. Choo (Eds.), *Computational Science and Its Applications - ICCSA 2006, Part III*. XXV, 1243 pages. 2006.

Vol. 3981: M. Gavrilova, O. Gervasi, V. Kumar, C.J. K. Tan, D. Taniar, A. Laganà, Y. Mun, H. Choo (Eds.), *Computational Science and Its Applications - ICCSA 2006, Part II*. XXVI, 1255 pages. 2006.

Vol. 3980: M. Gavrilova, O. Gervasi, V. Kumar, C.J. K. Tan, D. Taniar, A. Laganà, Y. Mun, H. Choo (Eds.), *Computational Science and Its Applications - ICCSA 2006, Part I*. LXXV, 1199 pages. 2006.

Vol. 3979: T.S. Huang, N. Sebe, M.S. Lew, V. Pavlović, M. Kölsch, A. Galata, B. Kisačanin (Eds.), *Computer Vision in Human-Computer Interaction*. XII, 121 pages. 2006.

- Vol. 3978: B. Hnich, M. Carlsson, F. Fages, F. Rossi (Eds.), Recent Advances in Constraints. VIII, 179 pages. 2006. (Sublibrary LNAI).
- Vol. 3976: F. Boavida, T. Plagemann, B. Stiller, C. Westphal, E. Monteiro (Eds.), Networking 2006. Networking Technologies, Services, and Protocols; Performance of Computer and Communication Networks; Mobile and Wireless Communications Systems. XXVI, 1276 pages. 2006.
- Vol. 3975: S. Mehrotra, D.D. Zeng, H. Chen, B.M. Thuraisingham, F.-Y. Wang (Eds.), Intelligence and Security Informatics. XXII, 772 pages. 2006.
- Vol. 3973: J. Wang, Z. Yi, J.M. Zurada, B.-L. Lu, H. Yin (Eds.), Advances in Neural Networks - ISNN 2006, Part III. XXIX, 1402 pages. 2006.
- Vol. 3972: J. Wang, Z. Yi, J.M. Zurada, B.-L. Lu, H. Yin (Eds.), Advances in Neural Networks - ISNN 2006, Part II. XXVII, 1444 pages. 2006.
- Vol. 3971: J. Wang, Z. Yi, J.M. Zurada, B.-L. Lu, H. Yin (Eds.), Advances in Neural Networks - ISNN 2006, Part I. LXVII, 1442 pages. 2006.
- Vol. 3970: T. Braun, G. Carle, S. Fahmy, Y. Koucheryav (Eds.), Wired/Wireless Internet Communications. XIV, 350 pages. 2006.
- Vol. 3968: K.P. Fishkin, B. Schiele, P. Nixon, A. Quigley (Eds.), Pervasive Computing. XV, 402 pages. 2006.
- Vol. 3967: D. Grigoriev, J. Harrison, E.A. Hirsch (Eds.), Computer Science - Theory and Applications. XVI, 684 pages. 2006.
- Vol. 3966: Q. Wang, D. Pfahl, D.M. Raffo, P. Wernick (Eds.), Software Process Change. XIV, 356 pages. 2006.
- Vol. 3965: M. Bernardo, A. Cimatti (Eds.), Formal Methods for Hardware Verification. VII, 243 pages. 2006.
- Vol. 3964: M. Ü. Uyar, A.Y. Duale, M.A. Fecko (Eds.), Testing of Communicating Systems. XI, 373 pages. 2006.
- Vol. 3963: O. Dikenelli, M.-P. Gleizes, A. Ricci (Eds.), Engineering Societies in the Agents World VI. X, 303 pages. 2006. (Sublibrary LNAI).
- Vol. 3962: W. IJsselstein, Y. de Kort, C. Midden, B. Eggen, E. van den Hoven (Eds.), Persuasive Technology. XII, 216 pages. 2006.
- Vol. 3960: R. Vieira, P. Quaresma, M.d.G.V. Nunes, N.J. Mamede, C. Oliveira, M.C. Dias (Eds.), Computational Processing of the Portuguese Language. XII, 274 pages. 2006. (Sublibrary LNAI).
- Vol. 3959: J.-Y. Cai, S. B. Cooper, A. Li (Eds.), Theory and Applications of Models of Computation. XV, 794 pages. 2006.
- Vol. 3958: M. Yung, Y. Dodis, A. Kiayias, T. Malkin (Eds.), Public Key Cryptography - PKC 2006. XIV, 543 pages. 2006.
- Vol. 3956: G. Barthe, B. Grégoire, M. Huisman, J.-L. Lanet (Eds.), Construction and Analysis of Safe, Secure, and Interoperable Smart Devices. IX, 175 pages. 2006.
- Vol. 3955: G. Antoniou, G. Potamias, C. Spyropoulos, D. Plexousakis (Eds.), Advances in Artificial Intelligence. XVII, 611 pages. 2006. (Sublibrary LNAI).
- Vol. 3954: A. Leonardis, H. Bischof, A. Pinz (Eds.), Computer Vision - ECCV 2006, Part IV. XVII, 613 pages. 2006.
- Vol. 3953: A. Leonardis, H. Bischof, A. Pinz (Eds.), Computer Vision - ECCV 2006, Part III. XVII, 649 pages. 2006.
- Vol. 3952: A. Leonardis, H. Bischof, A. Pinz (Eds.), Computer Vision - ECCV 2006, Part II. XVII, 661 pages. 2006.
- Vol. 3951: A. Leonardis, H. Bischof, A. Pinz (Eds.), Computer Vision - ECCV 2006, Part I. XXXV, 639 pages. 2006.
- Vol. 3950: J.P. Müller, F. Zambonelli (Eds.), Agent-Oriented Software Engineering VI. XVI, 249 pages. 2006.
- Vol. 3948: H.I. Christensen, H.-H. Nagel (Eds.), Cognitive Vision Systems. VIII, 367 pages. 2006.
- Vol. 3947: Y.-C. Chung, J.E. Moreira (Eds.), Advances in Grid and Pervasive Computing. XXI, 667 pages. 2006.
- Vol. 3946: T.R. Roth-Berghofer, S. Schulz, D.B. Leake (Eds.), Modeling and Retrieval of Context. XI, 149 pages. 2006. (Sublibrary LNAI).
- Vol. 3945: M. Hagiya, P. Wadler (Eds.), Functional and Logic Programming. X, 295 pages. 2006.
- Vol. 3944: J. Quiñonero-Candela, I. Dagan, B. Magnini, F. d'Alché-Buc (Eds.), Machine Learning Challenges. XIII, 462 pages. 2006. (Sublibrary LNAI).
- Vol. 3943: N. Guelfi, A. Savidis (Eds.), Rapid Integration of Software Engineering Techniques. X, 289 pages. 2006.
- Vol. 3942: Z. Pan, R. Aylett, H. Diener, X. Jin, S. Göbel, L. Li (Eds.), Technologies for E-Learning and Digital Entertainment. XXV, 1396 pages. 2006.
- Vol. 3941: S.W. Gilroy, M.D. Harrison (Eds.), Interactive Systems. XI, 267 pages. 2006.
- Vol. 3940: C. Saunders, M. Grobelnik, S. Gunn, J. Shawe-Taylor (Eds.), Subspace, Latent Structure and Feature Selection. X, 209 pages. 2006.
- Vol. 3939: C. Priami, L. Cardelli, S. Emmott (Eds.), Transactions on Computational Systems Biology IV. VII, 141 pages. 2006. (Sublibrary LNBI).
- Vol. 3936: M. Lalmas, A. MacFarlane, S. Rüger, A. Tombros, T. Tsikrika, A. Yavlinsky (Eds.), Advances in Information Retrieval. XIX, 584 pages. 2006.
- Vol. 3935: D. Won, S. Kim (Eds.), Information Security and Cryptology - ICISC 2005. XIV, 458 pages. 2006.
- Vol. 3934: J.A. Clark, R.F. Paige, F.A. C. Polack, P.J. Brooke (Eds.), Security in Pervasive Computing. X, 243 pages. 2006.
- Vol. 3933: F. Bonchi, J.-F. Boulicaut (Eds.), Knowledge Discovery in Inductive Databases. VIII, 251 pages. 2006.
- Vol. 3931: B. Apolloni, M. Marinaro, G. Nicosia, R. Tagliaferri (Eds.), Neural Nets. XIII, 370 pages. 2006.
- Vol. 3930: D.S. Yeung, Z.-Q. Liu, X.-Z. Wang, H. Yan (Eds.), Advances in Machine Learning and Cybernetics. XXI, 1110 pages. 2006. (Sublibrary LNAI).
- Vol. 3929: W. MacCaull, M. Winter, I. Düntsch (Eds.), Relational Methods in Computer Science. VIII, 263 pages. 2006.
- Vol. 3928: J. Domingo-Ferrer, J. Posegga, D. Schreckling (Eds.), Smart Card Research and Advanced Applications. XI, 359 pages. 2006.
- Vol. 3927: J. Hespanha, A. Tiwari (Eds.), Hybrid Systems: Computation and Control. XII, 584 pages. 2006.

Table of Contents

Invited Papers

Further Reflections on a Theory for Basic Algorithms <i>Allan Borodin</i>	1
Algorithmic DNA Self-assembly <i>Ming-Yang Kao</i>	10

Contributed Papers

Online Scheduling on Parallel Machines with Two GoS Levels <i>Yiwei Jiang</i>	11
Online Dial-A-Ride Problem with Time-Windows Under a Restricted Information Model <i>Fanglei Yi, Yinfeng Xu, Chunlin Xin</i>	22
Online Scheduling with Hard Deadlines on Parallel Machines <i>Jihuan Ding, Guochuan Zhang</i>	32
Maximizing the Throughput of Multiple Machines On-Line <i>Jae-Hoon Kim</i>	43
Lattice Embedding of Direction-Preserving Correspondence over Integrally Convex Set <i>Xi Chen, Xiaotie Deng</i>	53
Linear Programming Polytope and Algorithm for Mean Payoff Games <i>Ola Svensson, Sergei Vorobyov</i>	64
Atomic Routing Games on Maximum Congestion <i>Costas Busch, Malik Magdon-Ismail</i>	79
Equilibrium Distribution of Advertising Prices <i>Qianqin Chen, Wei-Guo Zhang, Guoliang Kuang</i>	92
Finding Faithful Boyce-Codd Normal Form Decompositions <i>Henning Koehler</i>	102

Instant Service Policy and Its Application to Deficit Round Robin <i>Jinoo Joung, Dongha Shin, Feifei Feng, Hongkyu Jeong</i>	114
A Compression-Boosting Transform for Two-Dimensional Data <i>Qiaofeng Yang, Stefano Lonardi, Avraham Melkman</i>	126
Non-metric Multicommodity and Multilevel Facility Location <i>Rudolf Fleischer, Jian Li, Shijun Tian, Hong Zhu</i>	138
Sublinear Time Width-Bounded Separators and Their Application to the Protein Side-Chain Packing Problem <i>Bin Fu, Zhixiang Chen</i>	149
Polygonal Curve Approximation Using Grid Points with Application to a Triangular Mesh Generation with Small Number of Different Edge Lengths <i>Shin-ichi Tanigawa, Naoki Katoh</i>	161
Distributions of Points and Large Convex Hulls of k Points <i>Hanno Lefmann</i>	173
Throwing Stones Inside Simple Polygons <i>Otfried Cheong, Hazel Everett, Hyo-Sil Kim, Sylvain Lazard, René Schott</i>	185
Some Basics on Tolerances <i>Boris Goldengorin, Gerold Jüger, Paul Molitor</i>	194
Note on a Class of Admission Control Policies for the Stochastic Knapsack Problem <i>Adriana F. Gabor, Jan-Kees C.W. van Ommeren</i>	207
Inverse Bottleneck Optimization Problems on Networks <i>Xiucui Guan, Jianzhong Zhang</i>	220
An Efficient Algorithm for Evacuation Problems in Dynamic Network Flows with Uniform Arc Capacity <i>Naoyuki Kamiyama, Naoki Katoh, Atsushi Takizawa</i>	231
Connected Set Cover Problem and Its Applications <i>Tian-Ping Shuai, Xiao-Dong Hu</i>	243
A Branch and Bound Algorithm for Exact, Upper, and Lower Bounds on Treewidth <i>Emgad H. Bachoore, Hans L. Bodlaender</i>	255

Recognition of Probe Cographs and Partitioned Probe Distance Hereditary Graphs <i>David B.Chandler, Maw-Shang Chang, Ton Kloks, Jiping Liu, Sheng-Lung Peng</i>	267
A New Approach for Solving the Maximum Clique Problem <i>P.J. Taillon</i>	279
The Approximability of the Exemplar Breakpoint Distance Problem <i>Zhixiang Chen, Bin Fu, Binhai Zhu</i>	291
Computing the λ -Seeds of a String <i>Qing Guo, Hui Zhang, Costas S. Iliopoulos</i>	303
Subsequence Packing: Complexity, Approximation, and Application <i>Minghui Jiang</i>	314
Decomposition Based Heuristic Approach to Frequency Reassignment Problem <i>Junghee Han</i>	324
Approximation Algorithms for Minimum Span Channel Assignment Problems <i>Yuichiro Miyamoto, Tomomi Matsui</i>	334
Weighted Broadcast in Linear Radio Networks <i>Gautam K. Das, Subhas C. Nandy</i>	343
Secure Overlay Network Design <i>Li (Erran) Li, Mohammad Mahdian, Vahab S. Mirrokni</i>	354
A Portfolio Selection Method Based on Possibility Theory <i>Wei-Guo Zhang, Qianqin Chen, Hai-Lin Lan</i>	367
Branch on Price: A Fast Winner Determination Algorithm for Discount Auctions <i>S. Kameshwaran, Lyés Benyoucef</i>	375
Note on an Auction Procedure for a Matching Game in Polynomial Time <i>Winfried Hochstättler, Hui Jin, Robert Nickel</i>	387
Author Index	395

Further Reflections on a Theory for Basic Algorithms

Allan Borodin

Department of Computer Science
University of Toronto
bor@cs.toronto.edu

1 Introduction

Can we optimally solve *Max2SAT* in (say) time $(|F| \log |F|)$ where $|F|$ is the length of formula F . Of course, since *Max2SAT* is *NP*-hard, we can confidently rely on our strongly held belief that no *NP*-hard problem can be solved optimally in polynomial time. But obtaining *unconditional* complexity lower bounds (even linear or near linear bounds) remains the central challenge of complexity theory. In the complementary fields of complexity theory and that of algorithm design and analysis, we ask questions such as “what is the best polynomial time approximation ratio” that can be achieved for *Max2SAT*. The best negative results are derived from the beautiful development of PCP proofs. In terms of obtaining better¹ approximation algorithms, we appeal to a variety of algorithmic techniques, including very basic techniques such as greedy algorithms, dynamic programming (with scaling), divide and conquer, local search and some more technically involved methods such as LP relaxation and randomized rounding, semi-definite programming (see [34] and [30] for an elegant presentation of these randomized methods and the concept of derandomization using conditional expectations). A more refined question might ask “what is the best approximation ratio (for a given problem such as *Max2SAT*) that can be obtained in (say) time $O(n \log n)$ ” where n is the length of the input in some standard representation of the problem. What algorithmic techniques should we consider if we are constrained to time $O(n \log n)$?

In order to bring some coherence to the “Design and Analysis of Algorithms”, most courses and texts will organize much of the content in terms of basic “algorithmic paradigms”, such as greedy algorithms, backtracking, dynamic programming, divide and conquer, local search, primal-dual, IP/LP rounding, etc. (but not etc. etc.). To this small set of paradigms, we can add randomization and sometimes very creative ways to utilize and combine these basic algorithmic approaches. Although we seem to be able to intuitively describe these basic classes of algorithms, we (in computer science) rarely attempt to *precisely* define what we mean by such terms as greedy, dynamic programming, etc. Clearly, a precise definition is required if we want to defend statements such as “there is no greedy

¹ For maximization (respectively, minimization) problems we will use approximation ratios ≤ 1 (resp. ≥ 1).

algorithm that provides a good approximation for problem X ” or “there is no efficient dynamic programming algorithm for optimally solving problem Y ”.

In the context of combinatorial search and optimization problems, I will suggest some simple but precise algorithmic models for some basic algorithmic paradigms. This is not a new approach, as there were (for example) a number of important attempts to characterize greedy algorithms (e.g. in terms of matroids [16] and greedoids [25]), and characterizing dynamic programming and branch and bound (e.g. in terms of formal languages [20]). In contrast to the elegant abstraction provided by matroids, we are not attempting to (say) characterize when *the* greedy algorithm is optimal for a set system but rather (similar, for example, to some previous studies for local search [24], “branch and bound algorithms” [12] and IP/LP rounding [4]) we are trying to explore the limitations of basic (simply defined) methods in terms of approximation ratios (or time complexity vs approximation/optimality results).

This talk is based on ideas and results from a number of recent papers. In particular, I will present formulations for greedy and greedy-like algorithms [8], simple dynamic programming and backtracking [10], and basic primal-dual/local ratio algorithms [7]. As will be explained, these models are all based on giving priorities to input items and concern worst case time complexity² for search and optimization problems.

2 Priority Algorithms as a Model of Greedy and Greedy-Like Algorithms

With the exception of naive brute force search, greedy algorithms are arguably the simplest approach³ for solving combinatorial optimization problems. The relation between *the* greedy algorithm for set systems and matroids was formalized by Rado [31] and Edmonds [16] with later extension to greedoids [25]. This early development did not address the use of greedy algorithms to achieve guaranteed approximation ratios but rather focused on the question of understanding when *the* greedy algorithm was optimal. Recently, k -extendible set systems are defined in [29] to help address the issue of when *the* greedy algorithm can provide a good approximation. In [8], we offered a simple model, called *priority algorithms* for greedy algorithms that can be applied in a wide variety of applications not restricted to set systems. We will briefly describe this model and some examples of well known greedy algorithms that are captured by this model. In the next section, we will use priority algorithms as the starting point for some other

² We note that the algorithmic models can be applied to any measure of complexity (e.g. space complexity, time vs space, average case or smoothed analysis).

³ On a conceptual level, local search algorithms are perhaps equally simple. Obviously, simplicity is in the eyes of the beholder and ignoring the complexity of optimally solving an LP relaxation, one can easily argue that IP/LP relaxations are also conceptually very simple. But whatever one’s experience and intuition, greedy algorithms are certainly considered to be conceptually simple.

“priority based” paradigms, namely “simple dynamic programming”, “simple backtracking” and “simple primal dual” algorithms.

The priority model (and the extensions that will follow) relies on the assumption that we represent an input instance as a set of “locally defined input items”, each item represented in some “natural” way. For a scheduling problem (such as interval or job scheduling to maximize profit, makespan minimization, etc.) the choice and representation of an “input item” is usually quite natural and not an issue. Namely, for scheduling problems, an input item is a “job” and each job is represented by the parameters of that job (e.g., the duration, deadline, value, etc. of the job). For other applications, such as graph theory there is a choice of whether to represent the items as edges or as vertices, and having done so there is a choice of how much information to provide within the representation of (say) a vertex. For the well known greedy Kruskal and Prim MST algorithms, the input items are edges, represented by their weights and their end points. For greedy vertex cover approximation algorithms the input items are the vertices, each vertex represented by its weight and its list of adjacent vertices⁴. Similarly, for the CNF-SAT problem, we can think of the clauses as the items or the propositional variables as the items. In the latter case, a variable could be represented by a full description of each clause in which it appears.

Of course there is no one “correct way” to define what is an input item and exactly how much “local information” should be included in the representation of an input item. For example, with regard to the interval selection problem, it seems that the most natural representation would be that each input interval is an item and is represented by a triple (s, f, v) where s (respectively, t and v) is the start (respectively, finish and value) of the interval. But one could also represent the input as an interval graph, or combining these representations by representing each interval I by the tuple (s, f, v, L) where L is a list of the intervals that intersect I . But, of course, this representation or the representation as an interval graph could result in a representation of size $\Omega(n^2)$ for an input instance having n intervals. This would seem to defeat the purpose of greedy algorithms which are utilized because of their efficiency.

After agreeing on the nature of the input representation, we are able to define a priority algorithm as formulated in [8]. The basic idea is that a priority algorithm is a *one-pass* algorithm in which the input is processed one input item at a time. The order or priority in which input items are “considered” is determined by a “local ordering”. When an input item is considered, the algorithm makes an irrevocable decision about this item. The nature of a problem usually determines the set of allowable decisions (e.g. accept/reject, schedule on particular machine, etc.). But what is an *allowable ordering* of the input items? We impose the priority condition that if inputs I_j and I_k are in the input sets $\mathcal{I}' \subseteq \mathcal{I}$ and I_j has higher priority than I_k in \mathcal{I} then that priority is maintained in the subset \mathcal{I}' . In particular then, any function $f : \mathcal{I} \rightarrow \mathbf{R}$ induces an allowable ordering by ordering input items in (say) non-decreasing (or non-increasing) order of their f

⁴ In fact, as discussed in [9], it is usually sufficient to represent a vertex by its list of adjacent edges.

value. Any function (including functions of arbitrarily high complexity or even non computable functions) that takes an input item given by its representation and produces a real number can use this real value as the priority of an item. The only other distinction to be made about the ordering is whether the ordering is *fixed* initially (before any item is considered) or if the ordering is *adaptive* in that the priority of an item can depend on the items previously considered.

In either the fixed or adaptive case, we emphasize that priority algorithms do not impose any explicit complexity limitations as the ordering and decisions being made can be of arbitrary complexity. It is only the “syntactic structure” of a priority algorithm that limits its power. Finally, we view priority algorithms as being “greedy-like” and reserve the term *greedy* for those priority algorithms which make “greedy (irrevocable) decisions” for each input item in the sense of making a decision as if this is the last input item and the decision must minimize/maximize the given objective function⁵.

We claim that the priority model seems to capture almost all algorithms that we commonly consider as greedy algorithms. The following are examples of fixed order priority algorithms: Kruskals MST, the maximal matching algorithm for unweighted vertex cover, optimal scheduling of unit profit intervals (ordering intervals according to non-decreasing finishing times), Graham’s “online” and LPT greedy approximation algorithms for minimizing makespan on identical machines. We also claim that for the exact *MaxkSAT* problem (where each clause has exactly k literals), the naive randomized algorithm (independently set each variable to true/false with probability $1/2$) can be derandomized to be a (online) priority algorithm (where the input items are the propositional variables represented by the description of the clauses in which they appear). This is similar to exercise 16.6 in [34] which indicates how to turn the naive randomized algorithm for MaxCut (independently place each vertex in S or \bar{S} with probability $1/2$) into an (online) greedy algorithm. The following are examples of adaptive order priority algorithms: the H_n approximation greedy set cover algorithm, Prim’s MST, Dijkstra’s shortest path algorithm (for digraphs where all edge costs are non-negative)⁶, Huffman optimal prefix trees⁷, various greedy algorithms for weighted vertex cover (see, for example, [13]). It also can be shown that the best (to date) polynomial time computable approximation ratio for the uncapacitated facility location problem is a greedy priority algorithm [27].

⁵ This “live for today” definition of greediness may not always make sense in settings where the input items are not “isolated”; for example, in a graph problem, if the items are the vertices, we may already know that a given vertex v is present since it is adjacent to a vertex already considered but v itself has not yet been considered. Our view is that in the context of approximation algorithms the distinction between greedy priority and (non-greedy) priority algorithms is not an essential distinction beyond the historical importance of the term. For optimal algorithms, each decision must be greedy by definition.

⁶ Here we view Dijkstra’s algorithm as computing the optimal tree from a single source to all other vertices.

⁷ Here we consider the input items to be nodes of the prefix tree, with leaf nodes representing the keys to be coded and internal nodes representing subtrees.

The priority algorithm model includes online algorithms (where the ordering of input items can be assumed to be dictated by an adversary) and as in the case of online algorithms one can derive negative results on approximation ratios (called the competitive ratio in the online setting). For example, it can be shown that weighted interval scheduling cannot have a c -approximation priority algorithm for any constant c (see [8]). Other negative results for priority algorithms can be found in [8, 32, 15, 3, 10, 9].

But have we captured everything that one would tend to call greedy? In [17], a 4-approximation algorithm called “greedy” is given for the weighted interval scheduling problem. The algorithm is a one pass algorithm in which rejections are irrevocable but acceptances are revocable. The condition that must be satisfied is that after each interval is considered, the partial solution being constructed is feasible. To incorporate such revocable acceptances, an appropriate extension of the priority model has been introduced and studied in [21].

3 Using Priorities Beyond Priority Algorithms

Obviously (and by design) the priority algorithm framework is a very limited algorithmic model. However, a number of other conceptually simple algorithms can also be viewed in terms of priorities given to input items. We briefly consider the stack model of [7], and the BT model of [10].

3.1 The Stack Model as a Model of Simple Primal Dual Algorithms

One of the most interesting developments in approximation algorithms has been the use of primal dual algorithms as pioneered in [1] and [18]. In many cases, primal dual algorithms can be realized as greedy algorithms (for example, as in the greedy approximation algorithms [23, 19] for the uncapacitated facility location problem) and also can be used to analyze known greedy algorithms (i.e. using the method of dual fitting). Simply stated, in a one pass primal dual algorithm, dual variables are increased until at least one constraint becomes tight. When each constraint corresponds to an input variable, then the fact that a constraint becomes tight can be used to make a decision about the corresponding variable. If our input representation has enough information to determine that a constraint has become tight then we can view the primal dual algorithm as an adaptive priority algorithm. For some applications of the primal dual method, a second clean up phase is also required (e.g. for covering problems to remove items not needed for a feasible solution and for packing problems to insure feasibility). The primal dual framework has been shown to be equivalent to the local ratio method [5, 6]. The local ratio method is used in [7] to motivate a stack model which attempts to model primal dual algorithms where the second clean up phase is a simple “popping” of a stack of items that were pushed onto the stack in the first phase (i.e. when dual constraints became satisfied). To be more precise, in a stack algorithm, the priority framework is used to decide on the order in which to consider input items and an “irrevocable accept/reject decision” is replaced by a decision to push the item onto the stack or else reject it. Once again, as in the