Symeon Bozapalidis
George Rahonis (Eds.)

LNCS 4728

Algebraic Informatics

Second International Conference, CAI 2007
Thessaloniki, Greece, May 2007
Revised Selected and Invited Papers

Springer

Symeon Bozapalidis George Rahonis (Eds.)

Algebraic Informatics

Second International Conference, CAI 2007
Thessaloniki, Greece, May 21-25, 2007
Revised Selected and Invited Papers

Springer

Volume Editors

Symeon Bozapalidis
George Rahonis
Aristotle University of Thessaloniki
54124 Thessaloniki, Greece
E-mail: {bozapali, grahonis}@math.auth.gr

Library of Congress Control Number: Applied for

CR Subject Classification (1998): F.3.1-2, F.4, D.2.1, I.1

LNCS Sublibrary: SL 1 – Theoretical Computer Science and General Issues

ISSN 0302-9743
ISBN-10 3-540-75413-X Springer Berlin Heidelberg New York
ISBN-13 978-3-540-75413-8 Springer Berlin Heidelberg New York

Springer is a part of Springer Science+Business Media

springer.com

Printed in Germany

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper SPIN: 12168668 06/3180 5 4 3 2 1 0

Lecture Notes in Computer Science 4728

Lecture Notes in Computer Science

Sublibrary 1: Theoretical Computer Science and General Issues

For information about Vols. 1–4528
please contact your bookseller or Springer

Vol. 4878: E. Tovar, P. Tsigas, H. Fouchal (Eds.), Principles of Distributed Systems. XIII, 457 pages. 2007.

Vol. 4873: S. Aluru, M. Parashar, R. Badrinath, V.K. Prasanna (Eds.), High Performance Computing – HiPC 2007. XXIV, 663 pages. 2007.

Vol. 4863: A. Bonato, F.R.K. Chung (Eds.), Algorithms and Models for the Web-Graph. X, 217 pages. 2007.

Vol. 4860: G. Eleftherakis, P. Kefalas, G. Păun, G. Rozenberg, A. Salomaa (Eds.), Membrane Computing. IX, 453 pages. 2007.

Vol. 4855: V. Arvind, S. Prasad (Eds.), FSTTCS 2007: Foundations of Software Technology and Theoretical Computer Science. XIV, 558 pages. 2007.

Vol. 4851: S. Boztaş, H.-F.(F.) Lu (Eds.), Applied Algebra, Algebraic Algorithms and Error-Correcting Codes. XII, 368 pages. 2007.

Vol. 4847: M. Xu, Y. Zhan, J. Cao, Y. Liu (Eds.), Advanced Parallel Processing Technologies. XIX, 767 pages. 2007.

Vol. 4846: I. Cervesato (Ed.), Advances in Computer Science – ASIAN 2007. XI, 313 pages. 2007.

Vol. 4838: T. Masuzawa, S. Tixeuil (Eds.), Stabilization, Safety, and Security of Distributed Systems. XIII, 409 pages. 2007.

Vol. 4835: T. Tokuyama (Ed.), Algorithms and Computation. XVII, 929 pages. 2007.

Vol. 4783: J. Holub, J. Žďárek (Eds.), Implementation and Application of Automata. XIII, 324 pages. 2007.

Vol. 4782: R. Perrott, B.M. Chapman, J. Subhlok, R.F. de Mello, L.T. Yang (Eds.), High Performance Computing and Communications. XIX, 823 pages. 2007.

Vol. 4771: T. Bartz-Beielstein, M.J. Blesa Aguilera, C. Blum, B. Naujoks, A. Roli, G. Rudolph, M. Sampels (Eds.), Hybrid Metaheuristics. X, 202 pages. 2007.

Vol. 4770: V.G. Ganzha, E.W. Mayr, E.V. Vorozhtsov (Eds.), Computer Algebra in Scientific Computing. XIII, 460 pages. 2007.

Vol. 4769: A. Brandstädt, D. Kratsch, H. Müller (Eds.), Graph-Theoretic Concepts in Computer Science. XIII, 341 pages. 2007.

Vol. 4763: J.-F. Raskin, P.S. Thiagarajan (Eds.), Formal Modeling and Analysis of Timed Systems. X, 369 pages. 2007.

Vol. 4746: A. Bondavalli, F. Brasileiro, S. Rajsbaum (Eds.), Dependable Computing. XV, 239 pages. 2007.

Vol. 4743: P. Thulasiraman, X. He, T.L. Xu, M.K. Denko, R.K. Thulasiram, L.T. Yang (Eds.), Frontiers of High Performance Computing and Networking ISPA 2007 Workshops. XXIX, 536 pages. 2007.

Vol. 4742: I. Stojmenovic, R.K. Thulasiram, L.T. Yang, W. Jia, M. Guo, R.F. de Mello (Eds.), Parallel and Distributed Processing and Applications. XX, 995 pages. 2007.

Vol. 4739: R. Moreno Díaz, F. Pichler, A. Quesada Arencibia (Eds.), Computer Aided Systems Theory – EUROCAST 2007. XIX, 1233 pages. 2007.

Vol. 4736: S. Winter, M. Duckham, L. Kulik, B. Kuipers (Eds.), Spatial Information Theory. XV, 455 pages. 2007.

Vol. 4732: K. Schneider, J. Brandt (Eds.), Theorem Proving in Higher Order Logics. IX, 401 pages. 2007.

Vol. 4731: A. Pelc (Ed.), Distributed Computing. XVI, 510 pages. 2007.

Vol. 4728: S. Bozapalidis, G. Rahonis (Eds.), Algebraic Informatics. VIII, 291 pages. 2007.

Vol. 4726: N. Ziviani, R. Baeza-Yates (Eds.), String Processing and Information Retrieval. XII, 311 pages. 2007.

Vol. 4719: R. Backhouse, J. Gibbons, R. Hinze, J. Jeuring (Eds.), Datatype-Generic Programming. XI, 369 pages. 2007.

Vol. 4711: C.B. Jones, Z. Liu, J. Woodcock (Eds.), Theoretical Aspects of Computing – ICTAC 2007. XI, 483 pages. 2007.

Vol. 4710: C.W. George, Z. Liu, J. Woodcock (Eds.), Domain Modeling and the Duration Calculus. XI, 237 pages. 2007.

Vol. 4708: L. Kučera, A. Kučera (Eds.), Mathematical Foundations of Computer Science 2007. XVIII, 764 pages. 2007.

Vol. 4707: O. Gervasi, M.L. Gavrilova (Eds.), Computational Science and Its Applications – ICCSA 2007, Part III. XXIV, 1205 pages. 2007.

Vol. 4706: O. Gervasi, M.L. Gavrilova (Eds.), Computational Science and Its Applications – ICCSA 2007, Part II. XXIII, 1129 pages. 2007.

Vol. 4705: O. Gervasi, M.L. Gavrilova (Eds.), Computational Science and Its Applications – ICCSA 2007, Part I. XLIV, 1169 pages. 2007.

Vol. 4703: L. Caires, V.T. Vasconcelos (Eds.), CONCUR 2007 – Concurrency Theory. XIII, 507 pages. 2007.

Vol. 4700: C.B. Jones, Z. Liu, J. Woodcock (Eds.), Formal Methods and Hybrid Real-Time Systems. XVI, 539 pages. 2007.

Vol. 4699: B. Kågström, E. Elmroth, J. Dongarra, J. Waśniewski (Eds.), Applied Parallel Computing. XXIX, 1192 pages. 2007.

Vol. 4698: L. Arge, M. Hoffmann, E. Welzl (Eds.), Algorithms – ESA 2007. XV, 769 pages. 2007.

Vol. 4697: L. Choi, Y. Paek, S. Cho (Eds.), Advances in Computer Systems Architecture. XIII, 400 pages. 2007.

Vol. 4688: K. Li, M. Fei, G.W. Irwin, S. Ma (Eds.), Bio-Inspired Computational Intelligence and Applications. XIX, 805 pages. 2007.

Vol. 4684: L. Kang, Y. Liu, S. Zeng (Eds.), Evolvable Systems: From Biology to Hardware. XIV, 446 pages. 2007.

Vol. 4683: L. Kang, Y. Liu, S. Zeng (Eds.), Advances in Computation and Intelligence. XVII, 663 pages. 2007.

Vol. 4681: D.-S. Huang, L. Heutte, M. Loog (Eds.), Advanced Intelligent Computing Theories and Applications. XXVI, 1379 pages. 2007.

Vol. 4672: K. Li, C. Jesshope, H. Jin, J.-L. Gaudiot (Eds.), Network and Parallel Computing. XVIII, 558 pages. 2007.

Vol. 4671: V.E. Malyshkin (Ed.), Parallel Computing Technologies. XIV, 635 pages. 2007.

Vol. 4669: J.M. de Sá, L.A. Alexandre, W. Duch, D. Mandic (Eds.), Artificial Neural Networks – ICANN 2007, Part II. XXXI, 990 pages. 2007.

Vol. 4668: J.M. de Sá, L.A. Alexandre, W. Duch, D. Mandic (Eds.), Artificial Neural Networks – ICANN 2007, Part I. XXXI, 978 pages. 2007.

Vol. 4666: M.E. Davies, C.J. James, S.A. Abdallah, M.D. Plumbley (Eds.), Independent Component Analysis and Blind Signal Separation. XIX, 847 pages. 2007.

Vol. 4665: J. Hromkovič, R. Královič, M. Nunkesser, P. Widmayer (Eds.), Stochastic Algorithms: Foundations and Applications. X, 167 pages. 2007.

Vol. 4664: J. Durand-Lose, M. Margenstern (Eds.), Machines, Computations, and Universality. X, 325 pages. 2007.

Vol. 4661: U. Montanari, D. Sannella, R. Bruni (Eds.), Trustworthy Global Computing. X, 339 pages. 2007.

Vol. 4649: V. Diekert, M.V. Volkov, A. Voronkov (Eds.), Computer Science – Theory and Applications. XIII, 420 pages. 2007.

Vol. 4647: R. Martin, M.A. Sabin, J.R. Winkler (Eds.), Mathematics of Surfaces XII. IX, 509 pages. 2007.

Vol. 4646: J. Duparc, T.A. Henzinger (Eds.), Computer Science Logic. XIV, 600 pages. 2007.

Vol. 4644: N. Azémard, L. Svensson (Eds.), Integrated Circuit and System Design. XIV, 583 pages. 2007.

Vol. 4641: A.-M. Kermarrec, L. Bougé, T. Priol (Eds.), Euro-Par 2007 Parallel Processing. XXVII, 974 pages. 2007.

Vol. 4639: E. Csuhaj-Varjú, Z. Ésik (Eds.), Fundamentals of Computation Theory. XIV, 508 pages. 2007.

Vol. 4638: T. Stützle, M. Birattari, H. H. Hoos (Eds.), Engineering Stochastic Local Search Algorithms. X, 223 pages. 2007.

Vol. 4630: H.J. van den Herik, P. Ciancarini, H.H.L.M.(J.) Donkers (Eds.), Computers and Games. XII, 283 pages. 2007.

Vol. 4628: L.N. de Castro, F.J. Von Zuben, H. Knidel (Eds.), Artificial Immune Systems. XII, 438 pages. 2007.

Vol. 4627: M. Charikar, K. Jansen, O. Reingold, J.D.P. Rolim (Eds.), Approximation, Randomization, and Combinatorial Optimization. XII, 626 pages. 2007.

Vol. 4624: T. Mossakowski, U. Montanari, M. Haveraaen (Eds.), Algebra and Coalgebra in Computer Science. XI, 463 pages. 2007.

Vol. 4623: M. Collard (Ed.), Ontologies-Based Databases and Information Systems. X, 153 pages. 2007.

Vol. 4621: D. Wagner, R. Wattenhofer (Eds.), Algorithms for Sensor and Ad Hoc Networks. XIII, 415 pages. 2007.

Vol. 4619: F. Dehne, J.-R. Sack, N. Zeh (Eds.), Algorithms and Data Structures. XVI, 662 pages. 2007.

Vol. 4618: S.G. Akl, C.S. Calude, M.J. Dinneen, G. Rozenberg, H.T. Wareham (Eds.), Unconventional Computation. X, 243 pages. 2007.

Vol. 4616: A.W.M. Dress, Y. Xu, B. Zhu (Eds.), Combinatorial Optimization and Applications. XI, 390 pages. 2007.

Vol. 4614: B. Chen, M. Paterson, G. Zhang (Eds.), Combinatorics, Algorithms, Probabilistic and Experimental Methodologies. XII, 530 pages. 2007.

Vol. 4613: F.P. Preparata, Q. Fang (Eds.), Frontiers in Algorithmics. XI, 348 pages. 2007.

Vol. 4600: H. Comon-Lundh, C. Kirchner, H. Kirchner (Eds.), Rewriting, Computation and Proof. XVI, 273 pages. 2007.

Vol. 4599: S. Vassiliadis, M. Bereković, T.D. Hämäläinen (Eds.), Embedded Computer Systems: Architectures, Modeling, and Simulation. XVIII, 466 pages. 2007.

Vol. 4598: G. Lin (Ed.), Computing and Combinatorics. XII, 570 pages. 2007.

Vol. 4596: L. Arge, C. Cachin, T. Jurdziński, A. Tarlecki (Eds.), Automata, Languages and Programming. XVII, 953 pages. 2007.

Vol. 4595: D. Bošnački, S. Edelkamp (Eds.), Model Checking Software. X, 285 pages. 2007.

Vol. 4590: W. Damm, H. Hermanns (Eds.), Computer Aided Verification. XV, 562 pages. 2007.

Vol. 4588: T. Harju, J. Karhumäki, A. Lepistö (Eds.), Developments in Language Theory. XI, 423 pages. 2007.

Vol. 4583: S.R. Della Rocca (Ed.), Typed Lambda Calculi and Applications. X, 397 pages. 2007.

Vol. 4580: B. Ma, K. Zhang (Eds.), Combinatorial Pattern Matching. XII, 366 pages. 2007.

Vol. 4576: D. Leivant, R. de Queiroz (Eds.), Logic, Language, Information and Computation. X, 363 pages. 2007.

Vol. 4547: C. Carlet, B. Sunar (Eds.), Arithmetic of Finite Fields. XI, 355 pages. 2007.

Vol. 4546: J. Kleijn, A. Yakovlev (Eds.), Petri Nets and Other Models of Concurrency – ICATPN 2007. XI, 515 pages. 2007.

Vol. 4545: H. Anai, K. Horimoto, T. Kutsia (Eds.), Algebraic Biology. XIII, 379 pages. 2007.

Vol. 4533: F. Baader (Ed.), Term Rewriting and Applications. XII, 419 pages. 2007.

￥424.00元

Preface

CAI 2007 was the 2nd International Conference on Algebraic Informatics. It was intended to cover the topics of algebraic semantics on graphs and trees, formal power series, syntactic objects, algebraic picture processing, infinite computation, acceptors and transducers for strings, trees, graphs, arrays, etc., and decision problems.

CAI 2007 was held during May 21–25, 2007 in Thessaloniki, Greece hosted by the Department of Mathematics of Aristotle University of Thessaloniki. The opening lecture was given by Jean Berstel and the other eight invited lectures by Jürgen Albert, Frank Drewes, Dora Giammarresi, Jozef Gruska, Jarkko Kari, Oliver Matz, Ulrike Prange (on behalf of Hartmut Ehrig), and Guo-Qiang Zhang. This volume contains eighth papers of the nine invited lectures and the accepted papers. We received 29 submissions, the contributors being from 14 countries. The Program Committee selected ten papers.

We are grateful to the members of the Program Committee for the evaluation of the submissions and the numerous referees who assisted in this work. We should like to thank all the contributors of CAI 2007 and especially the honorary guest and the invited speakers who kindly accepted our invitation to present their important work. Special thanks are due to Alfred Hofmann, the Editorial Director of LNCS, who gave us the opportunity to publish the proceedings of our conference in the LNCS series, as well as to Anna Kramer from Springer for the excellent cooperation. We are also grateful to the members of the Organizing Committee and a group of students who helped us with several organizing jobs. Last but not least we want to express our gratitude to Arto Salomaa for his constant interest in CAI and his support in Springer.

July 2007 Symeon Bozapalidis

Organization

CAI 2007 was organized by the Department of Mathematics, Aristotle University of Thessaloniki.

Program Committee

Symeon Bozapalidis, Thessaloniki (Chair)
Manfred Droste, Leipzig
Zoltan Ésik, Szeged/Tarragona
Werner Kuich, Vienna
Antonio Restivo, Palermo
Paul Spirakis, Patras
Heiko Vogler, Dresden

Organizing Committee

Symeon Bozapalidis (Chair)
Archontia Grammatikopoulou
Antonios Kalampakas
Costas Lolas
George Rahonis (Co-chair)

Referees

M. Bartha
S. Bozapalidis
R. Diaconescu
F. Drewes
M. Droste
Z. Ésik
P. Fiser
Z. Gazdag
R. Gentilini
A. Grammatikopoulou
A. Kalampakas
D. Kirsten
W. Kuich
D. Kuske
C. Lutz
A. Maletti
C. Mathissen
O. Matz
I. Maeurer
I. Meinecke
D. Mitsche
T. Mossakowski
F. Z. Nardelli
K. Ogata
A. Pluhar
A. Papistas
P. Pournara
I. Petre
G. Rahonis
A. Restivo
L. Rosaz
A. Sifaleras
P. Spirakis
Gh. Stefanescu
S. Vagvolgyi
H. Vogler
C. Umans

Table of Contents

Invited Papers

Contributed Papers

On Generalizations of Weighted Finite Automata and Graphics Applications

Jürgen Albert and German Tischler

Department of Computer Science, University of Würzburg
Am Hubland, D-97074 Würzburg, Germany
{albert,tischler}@informatik.uni-wuerzburg.de

Abstract. Already computations of ordinary finite automata can be interpreted as discrete grayscale or colour images. Input words are treated as addresses of pixel-components in a very natural way. In this well understood context already meaningful operations on images like zooming or self-similarity can be formally introduced. We will turn then to finite automata with states and transitions labeled by real numbers as weights. These Weighted Finite Automata (WFA), as introduced by Culik II, Karhumäki and Kari, have turned out to be powerful tools for image- and video-compression. The recursive inference-algorithm for WFA can exploit self-similarities within single pictures, between colour components and also in sequences of pictures. We will generalize WFA further to Parametric WFA by allowing different interpretations of the computed real vectors. These vector-components can be chosen as grayscale or colour intensities or e.g. as 3D-coordinates. Applications will be provided including well-known fractal sets and 3D polynomial spline-patches with textures.

1 Introduction

In standard textbooks on formal languages and automata theory the most common examples for finite automata deal with the analysis or transformation of strings, which appear as sequences of input symbols. Real world applications with finite automata are found e.g. in UNIX tools like grep, lex and many others. If it comes to more numerically motivated applications one can find e.g. counting modulo(k) for some given constant k, or the well-known finite machine over the input alphabet $\{(0,0),(0,1),(1,0),(1,1)\}$ adding two arbitrary long binary numbers from right to left. But the pumping-lemma also makes clear that the numerical capabilities of finite state devices are limited e.g. there is no finite machine computing correctly the product of two arbitrary binary numbers. But, as we will see in our example later on, it does not take drastic generalizations to achieve this by some simple weighted automaton.

Before introducing those Weighted Finite Automata we will relate finite acceptors with concepts from computer graphics (cf. [22]) like pixel-addressing, zooming, multi-resolution-properties, lossy compression etc. This should provide a clearer separation of the generalization steps to WFA and Parametric

S. Bozapalidis and G. Rahonis (Eds.): CAI 2007, LNCS 4728, pp. 1–22, 2007.

WFA (PWFA), which inherit much of their descriptive power already from the finite acceptors.

For the following we will only assume some basic knowledge about finite automata and elementary mathematics.

2 Finite Acceptors and Raster Images

We will start here with a minimalistic yet powerful approach, where the input-alphabet is always just $\Sigma = \{0, 1\}$, and the rasterized images consist only of either black or white picture elements. In our very first step we will even restrict ourselves to "1-dimensional images" embedded into the unit-interval.

2.1 Inputstrings as Addresses

For some given natural number $r \geq 0$ consider all strings in Σ^r, i.e. all binary strings of length r. In the 1-dimensional case we can associate an input word with a half-open interval:

$$x = b_1\ b_2\ \dots\ b_r,\ b_i \in \{0, 1\}$$
$$H(x) = [0.\ b_1\ b_2\ \dots\ b_r,\ 0.\ b_1\ b_2\ \dots\ b_r + 2^{-r})$$

of length 2^{-r} within the unit interval $[0, 1)$. This way the string 1011 stands for the interval $[\frac{11}{16}, \frac{12}{16})$. Increasing the length r of the strings by 1 therefore doubles the number of half-open sub-intervals of the unit interval.

Given any finite automaton A over Σ, some $r \geq 0$ and $x \in \Sigma^r$, we can assign the colour black to $H(x)$, iff x is accepted by A, $x \in L(A)$; otherwise the colour white is assigned to $H(x)$.

More formally, we assume the following representation for $A = (Q, \Sigma, M, I, F)$:

1. Q is a set of n states,
2. $\Sigma = \{0, 1\}$ is the binary input alphabet
3. $M = (M_0, M_1), M_i \in \{0, 1\}^{n \times n}$ are the transition matrices for the input-symbols 0, 1 resp. Here $M_i[s, t] = 1$ iff there is a transition from state s to state t labeled by input symbol i
4. $I \in \{0, 1\}^{n \times 1}$ is the initial vector. This is a row-vector (and a unit-vector), where the component for the start-state $I[s_0] = 1$, all others are 0
5. $F \in \{0, 1\}^{1 \times n}$ is the final column-vector, where a component $F[t] = 1$ iff the corresponding state t is a final state.

It should be obvious, that this notation is equivalent to the common definition of finite automata – i.e. finite state acceptors – if we declare acceptance for A as follows:

For $x = b_1\ b_2\ \dots\ b_r \in \{0, 1\}^r$ we have $x \in L(A)$ iff the function $f_A : \Sigma^* \to \mathbb{N}$ defined by

$$f_A(x) = I \times M_{b_1} \times M_{b_2} \cdots \times M_{b_r} \times F$$

yields some value $f_A(x) > 0$.

Fig. 1. Graph for A and input sequences of length 4

2.2 Image Generation by Finite Acceptors

Let us demonstrate the usefulness of this notation by the following finite automaton $A = (Q, \Sigma, M, I, F)$, where

1. $Q = \{q_1, q_2, q_3\}$,
2. $\Sigma = \{0, 1\}$,
3. the initial vector $I = (1, 0, 0)$,
4. the final vector $F^T = (1, 0, 0)$,
5. and for $M = (M_0,\ M_1)$ we have the transition matrices:

$$M_0 = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix}, M_1 = \begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}.$$

The language accepted by this automaton is $L(A) = \{00, 01, 10\}^*$, as can be seen easily from the transition-graph for A (cf. Fig. 1).

For the input string 1011 our function f_A yields $f_A(1011) = I \times M_1 \times M_0 \times M_1 \times M_1 \times F = 0$. By our convention, since 1011 is not accepted by A, the colour white is assigned to the interval $[\frac{11}{16}, \frac{12}{16})$. On the other hand the string 1000 is accepted, $f_A(1000) = I \times M_1 \times M_0 \times M_0 \times M_0 \times F = 1$, and $[\frac{8}{16}, \frac{9}{16})$ is painted black. (cf. Fig. 1).

If we look at the pattern of intervals for all words it is easy to see that for all odd lengths r of the input nothing is accepted and for $r = 0, 2, 4, 6, \ldots$ we can describe this informally as dividing all black intervals of stage r into four equal half-open parts and painting the last one white to arrive at stage $r + 2$.

This sounds of course very familiar if compared to the construction of the well-known Cantor set C, frequently called "Cantor dust" as well. Starting with the closed unit interval [0, 1] successively remove the middle thirds (as open intervals):

$$G_0 = [0, 1]$$

$$G_1 = [0, \frac{1}{3}] \cup [\frac{2}{3}, 1]$$

$$G_2 = [0, \frac{1}{9}] \cup [\frac{2}{9}, \frac{1}{3}] \cup [\frac{2}{3}, \frac{7}{9}] \cup [\frac{8}{9}, 1]$$

$$\ldots$$

Then the Cantor set C is defined as

$$C = \bigcap_{n=1}^{\infty} G_n,$$

where C is compact, has Lebesgue-measure 0 and frequently serves as a fundamental example of a fractal. So, it is no surprise that our example automaton A also shows fractal patterns. This will become even more apparent, if we change our interpretation of input words for A from 1-dimensional to 2-dimensional addresses.

2.3 Bi-level Images in 2D

The hierarchical form of addressing introduced above is generalized easily from the unit interval to the unit square $[0,1) \times [0,1)$, and further to any d-dimensional hypercube $[0,1)^d$, $d \geq 1$. The so-called Morton- or Z-order can achieve this desired hierarchical addressing in a very natural and intuitive way. Fig. 2 shows the numbering sequences for the unit square and address lengths of 2, 4 and 8.

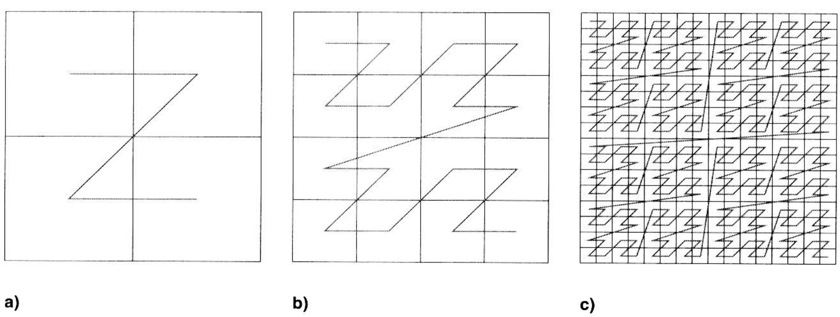


Fig. 2. Morton-Order (Z-Order)

In the common raster-scan-order pixels are arranged in a rectangular matrix and visited row-wise starting in the upper left corner. Thus, the Z-order at least matches with the raster-scan for the very first and last pixel. Depending on the applications it might be more favorable that the origin of the image is placed in the lower left corner e.g. to display graphs of functions or relations in the common way. Thus, the Morton-order might also become an N-order.

For our example of the finite automaton A above, it should be noted that we do not have to change anything within the definition or computation of the results to apply the 2D-interpretation to the addresses. Now the sequence 1011 leads into the white square $[\frac{3}{4}, 1) \times [\frac{1}{4}, \frac{1}{2})$ and analogously 1000 into the black square $[\frac{1}{2}, \frac{3}{4}) \times [0, \frac{1}{4})$. The corresponding pictures are given in Fig. 3 for the resolutions of $2^2 \times 2^2$ and $2^8 \times 2^8$ pixels. Thus, our example automaton generates the well-known Sierpinski-triangle.

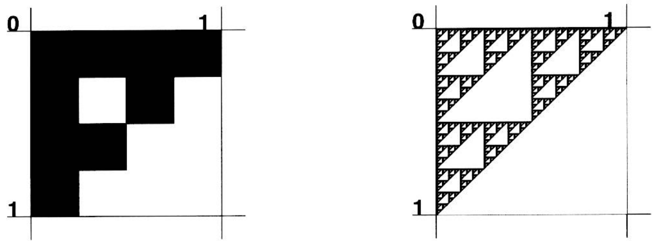
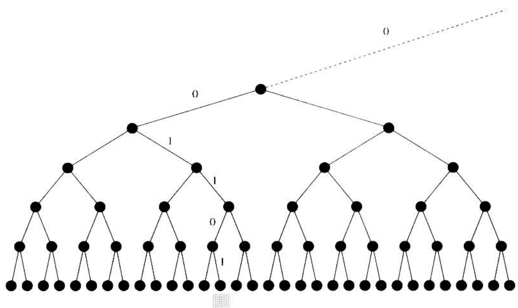


Fig. 3. Images generated by A of resolution 4×4 and 256×256



000000	000001	000100	000101	010000	010001	010100	010101
000010	000011	000110	000111	010010	010011	010110	010111
001000	001001	001100	001101	011000	011001	011100	011101
001010	001011	001110	001111	011010	011011	011110	011111
100000	100001	100100	100101	110000	110001	110100	110101
100010	100011	100110	100111	110010	110011	110110	110111
101000	101001	101100	101101	111000	111001	111100	111101
101010	101011	101110	101111	111010	111011	111110	111111

Fig. 4. Hierarchical addressing by bintrees

2.4 Bintrees for Addressing

In the 2D-interpretation an input-sequence of length $2r$ addresses a square of size $2^{-r} \times 2^{-r}$, and a sequence of length $2r+1$ then a rectangle of size $2^{-r-1} \times 2^{-r}$, as shown in Fig. 4 for $r = 6$.

The hierarchical and fractal nature of the Morton-order is apparent in representing the addresses in complete binary trees (or bintrees) of depth $2r$.

It is verified easily, that the hierarchical addressing works with the bit-wise interleaving of the x- and y-coordinates. Again, if we assume the origin of the image at the upper left corner, then any input address for a square of size $2^{-r} \times 2^{-r}$ like $w = y_1\, x_1\, y_2\, x_2 \ldots y_r\, x_r$ is found at the coordinates $x = x_1\, x_2 \ldots x_r$, $y = y_1\, y_2 \ldots y_r$.

Since each node of the bintree represents the image given by its leaves, zooming into this sub-image quite naturally can be associated with removing the common prefix-string u from all the pixel-addresses uv of that sub-image.

Returning to our example for the Sierpinski triangle we can convince ourselves also, that the concept of multi-resolution makes sense for images defined by finite acceptors. We can use the image of size $2^r \times 2^r$ pixels, scale it by one half in each dimension and place a copy into each of the three first sub-quadrants of the image with $2^{r+1} \times 2^{r+1}$ pixels. These steps of scaling and copying are used as an intuitive description for a basic form of fractal image generation (see [39]);

Fig. 5. Dithering and Inverting

similar approaches are found in [31], [32]; a theoretical discussion of the concepts can be found in [37].

The reader is invited to generate as an exercise variations of the Sierpinski triangle automaton for the diagonal line in the unit square and the upper left black triangle.

Some more detailed remarks are in order here: Though quadtrees, octtrees, etc. are found more frequently in literature for hierarchical addressing of higher-dimensional data ([3], [14]), bintrees have measurable advantages for image- and video-compression. The Morton-order is superior in general to the raster-scan addressing, since it can exploit spatial redundancies much better. The Hilbert-order, where only direct neighbour pixels are visited during the traversal, can be the preferred traversal-order, if spatial redundancies without directional bias are present in the raw data. It should be noted, that the Hilbert-order is self-similar and hierarchical too and can be mapped to the Morton-order by a simple finite state transduction. Many other variations of the traversal-order exist, like the Hilbert-Peano-order, the triangular or the circular order, which can serve for special applications.

Another meaningful operation for the graphical interpretation of regular languages is of course the complementation $\Sigma^* - L(A)$, which yields for any fixed resolution just the inverted pixel values.

The example displayed in Fig. 5 also shows the effects of dithering, which is frequently used in the printing processes to create the illusion of a greater colour-depth or – as in our case now – higher number of available grayness values.

2.5 Bit-Planes for Grayscale and Colour-Images

Whereas in the multi-resolution hierarchy for black and white images we can possibly see the same patterns in different sizes, we consider now short stacks of images of the same resolution. In many image formats which are frequently called "raw formats" each pixel-value with its components is stored in a fixed

Fig. 6. Most significant, second significant and least significant bit-plane for 512×512 grayscale image lena

number of bits. These are often 8 bits for gray and 24 for colour images or even 32 bits if transparency values are specified for the pixel positions. The red, green and blue component is usually coded in 8 bits then. Each of the bit-positions thus defines a separate bit-plane and if the bit sequences are intensity values coded in binary the positions range from the "Most Significant Bit", MSB, to the "Least Significant Bit", LSB.

Therefore, the whole image is representable as some stack of bilevel images (see Fig. 6). And since for any given image of finite resolution there is only a finite amount of bitplanes and bits to be coded, it is obvious, that finite state acceptors are sufficient in principle for the representation of common digital images. We can, for example, interpret the complete bintrees discussed above as transition graphs for those finite state acceptors. Starting with such a bintree for a bilevel image one can reduce the number of states by the classical state minimization algorithm, which will produce some directed acyclic graph (DAG). Remember that all accepted words are addresses of same length. Instead of starting out with complete bintrees it is usually better to begin with so-called region bintrees, where an inner node becomes a final node, if all the leaves in that subtree are final. For this region bintree again the state minimization algorithm can be applied. These approaches can be viewed as rough sketches for the WFA inference algorithm, where pictures consist of real-valued pixels and transitions in the finite automaton carry real-valued weights.

Several compression algorithms for the whole stack of bit-planes have been developed in the past for lossless and lossy image reproduction. For cartoon-like images with only a few different colours these can simply be variations of runlength-encoding or in the general case also sophisticated predictive methods, where considering several neighbouring bit-planes for encoding bitvalues can be employed as in the JBIG-standard ([27]). Lossy compression methods can take advantage of the fact that the bit-planes of the LSB or near to it mostly carry noise and can be neglected in the coding-process.

A totally different approach can be taken by following the contours of shapes in the style of turtle-graphics as picture defining languages. Even then many self-similarities occur, which can be exploited by WFA-variants, see [7], [33].