Brahim Hnich
Mats Carlsson
François Fages
Francesca Rossi (Eds.)

LNAI 3978

Recent Advances in Constraints

Joint ERCIM/CoLogNET International Workshop on Constraint Solving and Constraint Logic Programming, CSCLP 2005
Uppsala, Sweden, June 2005
Revised Selected and Invited Papers

Springer

Brahim Hnich Mats Carlsson
François Fages Francesca Rossi (Eds.)

Recent Advances in Constraints

Joint ERCIM/CoLogNET International Workshop
on Constraint Solving and Constraint Logic Programming
CSCLP 2005
Uppsala, Sweden, June 20-22, 2005
Revised Selected and Invited Papers

Springer

Volume Editors

Brahim Hnich
Izmir University of Economics
Faculty of Computer Science
Sakarya Caddesi No.156, 35330 Balcova, Izmir, Turkey
E-mail: brahim.hnich@ieu.edu.tr

Mats Carlsson
Swedish Institute of Computer Science, SICS
Uppsala Science Park, 75183 Uppsala, Sweden
E-mail: Mats.Carlsson@sics.se

François Fages
Institut National de Recherche en Informatique et en Automatique, INRIA
Domaine de Voluceau, Rocquencourt, 78153 Le Chesnay Cedex, France
E-mail: francois.fages@inria.fr

Francesca Rossi
University of Padova
Department of Pure and Applied Mathematics
Via G.B. Belzoni 7, 35131 Padova, Italy
E-mail: frossi@math.unipd.it

Library of Congress Control Number: 2006925094

CR Subject Classification (1998): I.2.3, F.3.1-2, F.4.1, D.3.3, F.2.2, G.1.6, I.2.8

LNCS Sublibrary: SL 7 – Artificial Intelligence

ISSN 0302-9743
ISBN-10 3-540-34215-X Springer Berlin Heidelberg New York
ISBN-13 978-3-540-34215-1 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: 11754602 06/3142 5 4 3 2 1 0

Lecture Notes in Artificial Intelligence 3978
Edited by J. G. Carbonell and J. Siekmann

Subseries of Lecture Notes in Computer Science

Preface

Constraints are a natural means of knowledge representation. This generality underpins the success with which constraint programming has been applied to a wide variety of disciplines in academia and industry such as production planning, communication networks, robotics, and bioinformatics.

This volume contains the extended and reviewed version of a selection of papers presented at the Joint ERCIM/CoLogNET International Workshop on Constraint Solving and Constraint Logic Programming (CSCLP 2005), which was held during June 20–22, 2005 in Uppsala, Sweden.

It also contains papers that were submitted in response to the open call that followed the workshop. The papers in this volume present research results regarding many aspects of constraint solving and constraint logic programming. This includes global constraints, search and heuristics, implementations of constraint systems, and a number of applications.

The editors would like to take the opportunity and thank all the authors who submitted a paper to this volume, as well as the reviewers for their helpful work.

This volume has been made possible thanks to the support of the European Research Consortium for Informatics and Mathematics (ERCIM), the European Network on Computational Logic (CoLogNET), the Swedish Institute of Computer Science (SICS), Science Foundation Ireland (Grant No. 00/PI.1/C075), and the Department of Information Science (DIS) at Uppsala University in Sweden.

We hope that the present volume is useful for anyone interested in the recent advances and new trends in constraint programming, constraint solving, problem modelling, and applications.

March 2006

B. Hnich, M. Carlsson, F. Fages, and F. Rossi
Organizers
CSCLP 2005

Organization

CSCLP 2005 was organized by the ERCIM Working Group on Constraints and the CoLogNET area on Logic and Constraint Logic Programming.

Organizing and Program Committee

Brahim Hnich	Izmir University of Economics, Turkey
Mats Carlsson	Swedish Institute of Computer Science, Sweden
François Fages	INRIA Rocquencourt, France
Francesca Rossi	University of Padova, Italy

Referees

O. Angelsmark
R. Barták
N. Beldiceanu
C. Bessiere
M.A. Carravilla
E. Coquery
T. Frühwirth
R. Haemmerlé
R. Hatamloo
E. Hebrard
Z. Kiziltan
A. Lodi
A. Meisels
I. Miguel
R. Martin
P. Moura
M.S. Pini
C-G. Quimper
I. Razgon
C. Ribeiro
C. Schulte
G. Schrader
T. Schrijvers
F. Spoto
G. Tack
S.A. Tarim
J. Thapper
W-J. van Hoeve
R.J. Wallace
T. Walsh
A. Wolf

Sponsoring Institutions

ERCIM Working Group on Constraints
European Network of Excellence CoLogNET
Science Foundation Ireland
Swedish Institute of Computer Science
Uppsala University (Department of Information Science)

Lecture Notes in Artificial Intelligence (LNAI)

Vol. 3978: B. Hnich, M. Carlsson, F. Fages, F. Rossi (Eds.), Recent Advances in Constraints. VIII, 179 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.

Vol. 3955: G. Antoniou, G. Potamias, C. Spyropoulos, D. Plexousakis (Eds.), Advances in Artificial Intelligence. XVII, 611 pages. 2006.

Vol. 3946: T.R. Roth-Berghofer, S. Schulz, D.B. Leake (Eds.), Modeling and Retrieval of Context. XI, 149 pages. 2006.

Vol. 3944: J. Quiñonero-Candela, I. Dagan, B. Magnini, F. d'Alché-Buc (Eds.), Machine Learning Challenges. XIII, 462 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.

Vol. 3918: W.K. Ng, M. Kitsuregawa, J. Li, K. Chang (Eds.), Advances in Knowledge Discovery and Data Mining. XXIV, 879 pages. 2006.

Vol. 3910: S.A. Brueckner, G.D.M. Serugendo, D. Hales, F. Zambonelli (Eds.), Engineering Self-Organising Systems. XII, 245 pages. 2006.

Vol. 3904: M. Baldoni, U. Endriss, A. Omicini, P. Torroni (Eds.), Declarative Agent Languages and Technologies III. XII, 245 pages. 2006.

Vol. 3900: F. Toni, P. Torroni (Eds.), Computational Logic in Multi-Agent Systems. XVII, 427 pages. 2006.

Vol. 3899: S. Frintrop, VOCUS: A Visual Attention System for Object Detection and Goal-Directed Search. XIV, 216 pages. 2006.

Vol. 3898: K. Tuyls, P.J. 't Hoen, K. Verbeeck, S. Sen (Eds.), Learning and Adaption in Multi-Agent Systems. X, 217 pages. 2006.

Vol. 3891: J.S. Sichman, L. Antunes (Eds.), Multi-Agent-Based Simulation VI. X, 191 pages. 2006.

Vol. 3890: S.G. Thompson, R. Ghanea-Hercock (Eds.), Defence Applications of Multi-Agent Systems. XII, 141 pages. 2006.

Vol. 3885: V. Torra, Y. Narukawa, A. Valls, J. Domingo-Ferrer (Eds.), Modeling Decisions for Artificial Intelligence. XII, 374 pages. 2006.

Vol. 3881: S. Gibet, N. Courty, J.-F. Kamp (Eds.), Gesture in Human-Computer Interaction and Simulation. XIII, 344 pages. 2006.

Vol. 3874: R. Missaoui, J. Schmidt (Eds.), Formal Concept Analysis. X, 309 pages. 2006.

Vol. 3873: L. Maicher, J. Park (Eds.), Charting the Topic Maps Research and Applications Landscape. VIII, 281 pages. 2006.

Vol. 3863: M. Kohlhase (Ed.), Mathematical Knowledge Management. XI, 405 pages. 2006.

Vol. 3862: R.H. Bordini, M. Dastani, J. Dix, A.E.F. Seghrouchni (Eds.), Programming Multi-Agent Systems. XIV, 267 pages. 2006.

Vol. 3849: I. Bloch, A. Petrosino, A.G.B. Tettamanzi (Eds.), Fuzzy Logic and Applications. XIV, 438 pages. 2006.

Vol. 3848: J.-F. Boulicaut, L. De Raedt, H. Mannila (Eds.), Constraint-Based Mining and Inductive Databases. X, 401 pages. 2006.

Vol. 3847: K.P. Jantke, A. Lunzer, N. Spyratos, Y. Tanaka (Eds.), Federation over the Web. X, 215 pages. 2006.

Vol. 3835: G. Sutcliffe, A. Voronkov (Eds.), Logic for Programming, Artificial Intelligence, and Reasoning. XIV, 744 pages. 2005.

Vol. 3830: D. Weyns, H. V.D. Parunak, F. Michel (Eds.), Environments for Multi-Agent Systems II. VIII, 291 pages. 2006.

Vol. 3817: M. Faundez-Zanuy, L. Janer, A. Esposito, A. Satue-Villar, J. Roure, V. Espinosa-Duro (Eds.), Nonlinear Analyses and Algorithms for Speech Processing. XII, 380 pages. 2006.

Vol. 3814: M. Maybury, O. Stock, W. Wahlster (Eds.), Intelligent Technologies for Interactive Entertainment. XV, 342 pages. 2005.

Vol. 3809: S. Zhang, R. Jarvis (Eds.), AI 2005: Advances in Artificial Intelligence. XXVII, 1344 pages. 2005.

Vol. 3808: C. Bento, A. Cardoso, G. Dias (Eds.), Progress in Artificial Intelligence. XVIII, 704 pages. 2005.

Vol. 3802: Y. Hao, J. Liu, Y.-P. Wang, Y.-m. Cheung, H. Yin, L. Jiao, J. Ma, Y.-C. Jiao (Eds.), Computational Intelligence and Security, Part II. XLII, 1166 pages. 2005.

Vol. 3801: Y. Hao, J. Liu, Y.-P. Wang, Y.-m. Cheung, H. Yin, L. Jiao, J. Ma, Y.-C. Jiao (Eds.), Computational Intelligence and Security, Part I. XLI, 1122 pages. 2005.

Vol. 3789: A. Gelbukh, Á. de Albornoz, H. Terashima-Marín (Eds.), MICAI 2005: Advances in Artificial Intelligence. XXVI, 1198 pages. 2005.

Vol. 3782: K.-D. Althoff, A. Dengel, R. Bergmann, M. Nick, T.R. Roth-Berghofer (Eds.), Professional Knowledge Management. XXIII, 739 pages. 2005.

Vol. 3763: H. Hong, D. Wang (Eds.), Automated Deduction in Geometry. X, 213 pages. 2006.

Vol. 3755: G.J. Williams, S.J. Simoff (Eds.), Data Mining. XI, 331 pages. 2006.

Vol. 3735: A. Hoffmann, H. Motoda, T. Scheffer (Eds.), Discovery Science. XVI, 400 pages. 2005.

Vol. 3734: S. Jain, H.U. Simon, E. Tomita (Eds.), Algorithmic Learning Theory. XII, 490 pages. 2005.

Vol. 3721: A.M. Jorge, L. Torgo, P.B. Brazdil, R. Camacho, J. Gama (Eds.), Knowledge Discovery in Databases: PKDD 2005. XXIII, 719 pages. 2005.

Vol. 3720: J. Gama, R. Camacho, P.B. Brazdil, A.M. Jorge, L. Torgo (Eds.), Machine Learning: ECML 2005. XXIII, 769 pages. 2005.

Vol. 3717: B. Gramlich (Ed.), Frontiers of Combining Systems. X, 321 pages. 2005.

Vol. 3702: B. Beckert (Ed.), Automated Reasoning with Analytic Tableaux and Related Methods. XIII, 343 pages. 2005.

Vol. 3698: U. Furbach (Ed.), KI 2005: Advances in Artificial Intelligence. XIII, 409 pages. 2005.

Vol. 3690: M. Pěchouček, P. Petta, L.Z. Varga (Eds.), Multi-Agent Systems and Applications IV. XVII, 667 pages. 2005.

Vol. 3684: R. Khosla, R.J. Howlett, L.C. Jain (Eds.), Knowledge-Based Intelligent Information and Engineering Systems, Part IV. LXXIX, 933 pages. 2005.

Vol. 3683: R. Khosla, R.J. Howlett, L.C. Jain (Eds.), Knowledge-Based Intelligent Information and Engineering Systems, Part III. LXXX, 1397 pages. 2005.

Vol. 3682: R. Khosla, R.J. Howlett, L.C. Jain (Eds.), Knowledge-Based Intelligent Information and Engineering Systems, Part II. LXXIX, 1371 pages. 2005.

Vol. 3681: R. Khosla, R.J. Howlett, L.C. Jain (Eds.), Knowledge-Based Intelligent Information and Engineering Systems, Part I. LXXX, 1319 pages. 2005.

Vol. 3673: S. Bandini, S. Manzoni (Eds.), AI*IA 2005: Advances in Artificial Intelligence. XIV, 614 pages. 2005.

Vol. 3662: C. Baral, G. Greco, N. Leone, G. Terracina (Eds.), Logic Programming and Nonmonotonic Reasoning. XIII, 454 pages. 2005.

Vol. 3661: T. Panayiotopoulos, J. Gratch, R. Aylett, D. Ballin, P. Olivier, T. Rist (Eds.), Intelligent Virtual Agents. XIII, 506 pages. 2005.

Vol. 3658: V. Matoušek, P. Mautner, T. Pavelka (Eds.), Text, Speech and Dialogue. XV, 460 pages. 2005.

Vol. 3651: R. Dale, K.-F. Wong, J. Su, O.Y. Kwong (Eds.), Natural Language Processing – IJCNLP 2005. XXI, 1031 pages. 2005.

Vol. 3642: D. Ślęzak, J. Yao, J.F. Peters, W. Ziarko, X. Hu (Eds.), Rough Sets, Fuzzy Sets, Data Mining, and Granular Computing, Part II. XXIII, 738 pages. 2005.

Vol. 3641: D. Ślęzak, G. Wang, M. Szczuka, I. Düntsch, Y. Yao (Eds.), Rough Sets, Fuzzy Sets, Data Mining, and Granular Computing, Part I. XXIV, 742 pages. 2005.

Vol. 3635: J.R. Winkler, M. Niranjan, N.D. Lawrence (Eds.), Deterministic and Statistical Methods in Machine Learning. VIII, 341 pages. 2005.

Vol. 3632: R. Nieuwenhuis (Ed.), Automated Deduction – CADE-20. XIII, 459 pages. 2005.

Vol. 3630: M.S. Capcarrère, A.A. Freitas, P.J. Bentley, C.G. Johnson, J. Timmis (Eds.), Advances in Artificial Life. XIX, 949 pages. 2005.

Vol. 3626: B. Ganter, G. Stumme, R. Wille (Eds.), Formal Concept Analysis. X, 349 pages. 2005.

Vol. 3625: S. Kramer, B. Pfahringer (Eds.), Inductive Logic Programming. XIII, 427 pages. 2005.

Vol. 3620: H. Muñoz-Ávila, F. Ricci (Eds.), Case-Based Reasoning Research and Development. XV, 654 pages. 2005.

Vol. 3614: L. Wang, Y. Jin (Eds.), Fuzzy Systems and Knowledge Discovery, Part II. XLI, 1314 pages. 2005.

Vol. 3613: L. Wang, Y. Jin (Eds.), Fuzzy Systems and Knowledge Discovery, Part I. XLI, 1334 pages. 2005.

Vol. 3607: J.-D. Zucker, L. Saitta (Eds.), Abstraction, Reformulation and Approximation. XII, 376 pages. 2005.

Vol. 3601: G. Moro, S. Bergamaschi, K. Aberer (Eds.), Agents and Peer-to-Peer Computing. XII, 245 pages. 2005.

Vol. 3600: F. Wiedijk (Ed.), The Seventeen Provers of the World. XVI, 159 pages. 2006.

Vol. 3596: F. Dau, M.-L. Mugnier, G. Stumme (Eds.), Conceptual Structures: Common Semantics for Sharing Knowledge. XI, 467 pages. 2005.

Vol. 3593: V. Mařík, R. W. Brennan, M. Pěchouček (Eds.), Holonic and Multi-Agent Systems for Manufacturing. XI, 269 pages. 2005.

Vol. 3587: P. Perner, A. Imiya (Eds.), Machine Learning and Data Mining in Pattern Recognition. XVII, 695 pages. 2005.

Vol. 3584: X. Li, S. Wang, Z.Y. Dong (Eds.), Advanced Data Mining and Applications. XIX, 835 pages. 2005.

Vol. 3581: S. Miksch, J. Hunter, E.T. Keravnou (Eds.), Artificial Intelligence in Medicine. XVII, 547 pages. 2005.

Vol. 3577: R. Falcone, S. Barber, J. Sabater-Mir, M.P. Singh (Eds.), Trusting Agents for Trusting Electronic Societies. VIII, 235 pages. 2005.

Vol. 3575: S. Wermter, G. Palm, M. Elshaw (Eds.), Biomimetic Neural Learning for Intelligent Robots. IX, 383 pages. 2005.

Vol. 3571: L. Godo (Ed.), Symbolic and Quantitative Approaches to Reasoning with Uncertainty. XVI, 1028 pages. 2005.

Vol. 3559: P. Auer, R. Meir (Eds.), Learning Theory. XI, 692 pages. 2005.

Vol. 3558: V. Torra, Y. Narukawa, S. Miyamoto (Eds.), Modeling Decisions for Artificial Intelligence. XII, 470 pages. 2005.

Vol. 3554: A.K. Dey, B. Kokinov, D.B. Leake, R. Turner (Eds.), Modeling and Using Context. XIV, 572 pages. 2005.

Vol. 3550: T. Eymann, F. Klügl, W. Lamersdorf, M. Klusch, M.N. Huhns (Eds.), Multiagent System Technologies. XI, 246 pages. 2005.

Vol. 3539: K. Morik, J.-F. Boulicaut, A. Siebes (Eds.), Local Pattern Detection. XI, 233 pages. 2005.

Vol. 3538: L. Ardissono, P. Brna, A. Mitrović (Eds.), User Modeling 2005. XVI, 533 pages. 2005.

Vol. 3533: M. Ali, F. Esposito (Eds.), Innovations in Applied Artificial Intelligence. XX, 858 pages. 2005.

Table of Contents

Global Constraints

Search and Heuristics

Language and Implementation Issues

Modeling

The All Different and Global Cardinality Constraints on Set, Multiset and Tuple Variables

Claude-Guy Quimper[1] and Toby Walsh[2]

[1] School of Computer Science, University of Waterloo, Canada
cquimper@math.uwaterloo.ca
[2] NICTA and UNSW, Sydney, Australia
tw@cse.unsw.edu.au

Abstract. We describe how the propagator for the ALL-DIFFERENT constraint can be generalized to prune variables whose domains are not just simple finite domains. We show, for example, how it can be used to propagate set variables, multiset variables and variables which represent tuples of values. We also describe how the propagator for the global cardinality constraint (which is a generalization of the ALL-DIFFERENT constraint) can be generalized in a similar way. Experiments show that such propagators can be beneficial in practice, especially when the domains are large.

1 Introduction

Constraint programming has restricted itself largely to finding values for variables taken from given finite domains. However, we might want to consider variables whose values have more structure. We might, for instance, want to find a set of values for a variable [12, 13, 14, 15], a multiset of values for a variable [16], an ordered tuple of values for a variable, or a string of values for a variable. There are a number of reasons to want to enrich the type of values taken by a variable. First, we can reduce the space needed to represent possible domain values. For example, we can represent the exponential number of subsets for a set variable with just an upper and lower bound representing possible and definite elements in the set. Second, we can improve the efficiency of constraint propagators for such variables by exploiting the structure in the domain. For example, it might be sufficient to consider each of the possible elements in a set in turn, rather than the exponential number of subsets. Third, we inherit all the usual benefits of data abstraction like ease of debugging and code maintenance.

As an example, consider the round robin sports scheduling problem (prob026 in CSPLib). In this problem, we wish to find a game for each slot in the schedule. Each game is a pair of teams. There are a number of constraints that the schedule needs to satisfy including that all games are different from each other. We therefore would like a propagator which works on an ALL-DIFFERENT constraint posted on variables whose values are pairs (binary tuples). In this paper, we consider how to implement such constraints efficiently and effectively. We show how two of the most important constraint propagators, those for the ALL-DIFFERENT

B. Hnich et al. (Eds.): CSCLP 2005, LNAI 3978, pp. 1–13, 2006.

and the global cardinality constraint (*gcc*) can be extended to deal with variables whose values are sets, multisets or tuples.

2 Propagators for the All-Different Constraint

Propagating the All-Different constraint consists of detecting the values in the variable domains that cannot be part of an assignment satisfying the constraint. To design his propagator, Leconte [18] introduced the concept of *Hall set* based on Hall's work [1].

Definition 1. *A* Hall set *is a set H of values such that the number of variables whose domain is contained in H is equal to the cardinality of H. More formally, H is a Hall set if and only if $|H| = |\{x_i \mid dom(x_i) \subseteq H\}|$.*

Consider the following example.

Example 1. Let $\text{dom}(x_1) = \{3, 4\}$, $\text{dom}(x_2) = \{3, 4\}$, and $\text{dom}(x_3) = \{2, 4, 5\}$ be three variable domains subject to an All-Different constraint. The set $H = \{3, 4\}$ is a Hall set since it contains two elements and the two variable domains $\text{dom}(x_1)$ and $\text{dom}(x_2)$ are contained in H.

In Example 1, variables x_1 and x_2 must be assigned to values 3 and 4, making these two values unavailable for other variables. Therefore, value 4 should be removed from the domain of x_3.

To enforce domain consistency, it is necessary and sufficient to detect every Hall set H and remove its values from the domains that are not fully contained in H. This is exactly what Régin's propagator [4] does using matching theory to detect Hall sets. Leconte [18], Puget [20], López-Ortiz et al. [19] use simpler ways to detect Hall intervals in order to achieve weaker consistencies.

3 Beyond Integer Variables

A propagator designed for integer variables can be applied to any type of variable whose domain can be enumerated. For instance, let the following variables be sets whose domains are expressed by a set of required values and a set of allowed values.

$$\{\} \subseteq S_1, S_2, S_3, S_4 \subseteq \{1, 2\} \text{ and } \{\} \subseteq S_5, S_6 \subseteq \{2, 3\}$$

Variable domains can be expanded as follows:

$$S_1, S_2, S_3, S_4 \in \{\{\}, \{1\}, \{2\}, \{1, 2\}\} \text{ and } S_5, S_6 \in \{\{\}, \{2\}, \{3\}, \{2, 3\}\}$$

And then by enforcing GAC on the All-Different constraint, we obtain

$$S_1, S_2, S_3, S_4 \in \{\{\}, \{1\}, \{2\}, \{1, 2\}\} \text{ and } S_5, S_6 \in \{\{3\}, \{2, 3\}\}$$

We can now convert the domains back to their initial representation.

$$\{\} \subseteq S_1, S_2, S_3, S_4 \subseteq \{1, 2\} \text{ and } \{3\} \subseteq S_5, S_6 \subseteq \{2, 3\}$$

This technique always works but is not tractable in general since variable domains might have exponential size. For instance, the domain of $\{\} \subseteq S_i \subseteq \{1, \ldots, n\}$ contains 2^n elements. The following important lemma allows us to ignore such variables and focus just on those with "small" domains.

Lemma 1. *Let n be the number of variables and let F be a set of variables whose domains are not contained in any Hall set. Let $x_i \notin F$ be a variable whose domain contains more than $n - |F|$ values. Then $dom(x_i)$ is not contained in any Hall set.*

Proof. The largest Hall set can contain the domain of $n - |F|$ variables and therefore has at most $n - |F|$ values. If $|\text{dom}(x_i)| > n - |F|$, then $\text{dom}(x_i)$ cannot be contained in any Hall set. □

Using Lemma 1, we can iterate through the variables and append to a set F those whose domain cannot be contained in a Hall set. A propagator for the ALL-DIFFERENT constraint can prune the domains not in F and find all Hall sets. Values in Hall sets can then be removed from the variable domains in F. This technique ensures that domains larger than n do not slow down the propagation. Algorithm 1 exhibits the process for a set of (possibly non-integer) variables X.

Algorithm 1. ALL-DIFFERENT propagator for variables with large domains

$F \leftarrow \emptyset$
Sort variables such that $|\text{dom}(x_i)| \geq |\text{dom}(x_{i+1})|$
for $x_i \in X$ **do**
1   **if** $|dom(x_i)| > n - |F|$ **then** $F \leftarrow F \cup \{x_i\}$
2 Expand domains of variables in $X - F$.
Find values H belonging to a Hall set and propagate the All-Different constraint on variables $X - F$.
for $x_i \in F$ **do**
    $\text{dom}(x_i) \leftarrow \text{dom}(x_i) - H$;
3 Collapse domains of variables in $X - F$.

To apply our new techniques, three conditions must be satisfied by the representation of the variables:

1. Computing the size of the domain must be tractable (Line 1).
2. Domains must be efficiently enumerable (Line 2).
3. Domains must be efficiently computed from an enumeration of values (Line 3).

The next sections describe how different representations of domains for set, multiset and tuple variables can meet these three conditions.

4 All-Different on Sets

Several representations of domains have been suggested for set variables. We show how their cardinality can be computed and their domain enumerated efficiently. One of the most common representations for a set are the required elements lb and the allowed elements ub, with any set S satisfying $lb \subseteq S \subseteq ub$ belongs to the domain [12, 14]. The number of sets in the domain is given by $2^{|ub-lb|}$. We can enumerate all these sets simply by enumerating all subsets of $ub - lb$ and adding them to the elements from lb. A set can be represented as a binary vector where each element is associated to a bit. A bit equals 1 if its corresponding element is in the set and equals 0 if its corresponding element is not in the set. Enumerating all subsets of $ub - lb$ is reduced to the problem of enumerating all binary vectors between 0 and $2^{|ub-lb|}$ exclusively which can be done in $O(2^{|ub-lb|})$ steps, i.e. $O(|\text{dom}(S_i)|)$ steps.

In order to exclude from the domain undesired sets, one can also add a cardinality variable [3]. The domain of a set variable is therefore expressed by $\text{dom}(S_i) = \{S \mid lb \subseteq S \subseteq ub, |S| \in \text{dom}(C)\}$ where C is an integer variable. We assume that C is consistent with lb and ub, i.e. $\min(C) >= |lb|$ and $\max(C) <= |ub|$. The size of the domain is given by Equation 1 where $\binom{a}{b}$ is the binomial coefficient.

$$|\text{dom}(S_i)| = \sum_{j \in C} \binom{|ub - lb|}{j - |lb|} \tag{1}$$

The binomial coefficients can efficiently be computed as explained in Chapter 6.1 of [10]. The identity $\binom{n}{k+1} = \frac{n-k}{k+1}\binom{n}{k}$ can be particularly useful to compute the summation when the domain of C is an interval. The number of steps required to compute $|\text{dom}(S_i)|$ is bounded by $O(|\text{dom}(C)|)$.

Algorithm 2 enumerates all combinations of t elements chosen from elements 0 to $n-1$. Each element i in a combination is mapped to the i^{th} element in $ub - lb$. By enumerating all t-combinations for $t \in \text{dom}(C)$ to which we add the required elements lb, we enumerate all sets in $|\text{dom}(S_i)|$. Algorithm 2 has a time complexity of $O(t + \binom{n}{t})$. Since we call it for each $t \in \text{dom}(C)$, the total time complexity simplifies to $O(\max(|ub - lb|, |\text{dom}(S_i)|))$.

Sadler and Gervet [7] suggest adding a lexicographic ordering constraint to the domain description. This gives more expressiveness to the domain representation and can eliminate more undesired sets. that We say that $S_1 < S_2$ holds if S_1 comes before S_2 in a lexicographical order. The new domain representation now involves two lexicographic bounds l and u.

$$\text{dom}(S_i) = \{S \mid lb \subseteq S \subseteq ub, |S| = C, l \leq S \leq u\} \tag{2}$$

Knuth [8] represents all subsets of a set using a binomial tree like the one in Figure 1. The empty set is the root of the tree to which we can add elements by branching to a child. One can list all sets in lexicographical order by visiting

Algorithm 2. Enumerate the $\binom{n}{t}$ combinations of t elements between 0 and $n-1$. (Source: Algorithm T, Knuth [8] p.5)

$c_j \leftarrow j-1, \forall j\ 1 \leq j \leq t$
$c_{t+1} \leftarrow n$
$c_{t+2} \leftarrow 0$
repeat
  visit $c_t, c_{t-1}, \ldots, c_1$
  $j \leftarrow 1$
  while $c_j + 1 = c_{j+1}$ **do**
    $c_j \leftarrow j-1$
    $j \leftarrow j+1$
  $c_j \leftarrow c_j + 1$
until $j > t$

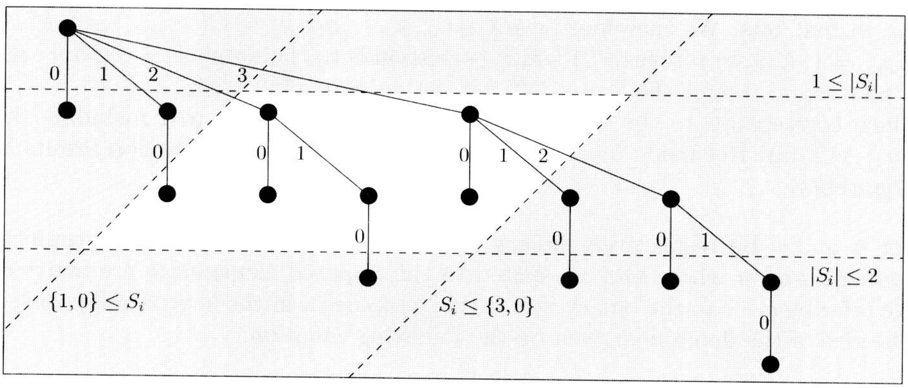

Fig. 1. Binomial tree representing the domain $\emptyset \subseteq S_i \subseteq \{0,1,2,3\}$, $1 \leq |S_i| \leq 2$, and $\{1,0\} \leq S_i \leq \{3,0\}$

the tree from left to right with a depth-first-search (DFS). We clearly see that the lexicographic constraints are orthogonal to the cardinality constraints.

Based on the binomial tree, we compute, level by level, the number of sets that belong to the domain. Notice that sets at level k have cardinality k. A set in the variable domain can be encoded with a binary vector of size $|ub - lb|$ where each bit is associated to a potential element in $ub - lb$. A bit set to one indicates the element belongs to the set while a bit set to zero means that the element does not belong to the set. The number of sets of cardinality k in the domain is equal to the number of binary vectors with k bits set to one and that lexicographically lie between l and u. Let $[u_m, \ldots, u_1]$ be the binary representation of the lexicographic upper bound u. Assuming $\binom{b}{a} = 0$ for all negative values of a, function $C([u_m, \ldots, u_1], k)$ returns the number of binary vectors that are lexicographically smaller than or equal to u and that have k bits set to one.

$$C([s_m, \ldots, s_1], k) = \sum_{i=1}^{m} s_i \binom{i-1}{k - \sum_{j=i+1}^{m} s_j} + \delta(\boldsymbol{s}, k) \quad (3)$$

$$\delta([s_m, \ldots, s_1], k) = \begin{cases} 1 & \text{if } \sum_{i=1}^{m} s_i = k \text{ and } s_0 = 0 \\ 0 & \text{otherwise} \end{cases} \quad (4)$$

Lemma 2. *Equation 3 is correct.*

Proof. We prove correctness by induction on m. For $m = 1$, Equation 3 holds with both $k = 0$ and $k = 1$. Suppose the equation holds for m, we want to prove it also holds for $m + 1$. We have

$$C([s_{m+1}, \ldots, s_1], k) = s_{m+1}\binom{m}{k} + C([s_m, \ldots, s_1], k - s_{m+1}) \quad (5)$$

If $s_{m+1} = 0$, the lexicographic constraint is the same as if we only consider the m first bits. We therefore have $C([s_{m+1}, \ldots, s_1], k) = C([s_m, \ldots, s_1], k)$. If $s_{m+1} = 1$, $C(s, k)$ returns $\binom{m}{k}$ which corresponds to the number of vectors with k bits set to 1 and the $(m + 1)^{th}$ bit set to zero plus $C([s_m, \ldots, s_1], k - 1)$ which corresponds to the number of vectors with k bits set to 1 including the $(m + 1)^{th}$ bit. Recursion 5 is therefore correct. Solving this recursion results in Equation 3. □

Let a and b be respectively binary vectors associated to the lexicographical bounds l and u where bits associated to the required elements lb are omitted. We refer by $a - 1$ to the binary vector that precedes a in the lexicographic order. The size of the domain is given by the following equation.

$$|\text{dom}(S_i)| = \sum_{k \in C} (C(b, k) - C(a - 1, k))$$

Function C can be evaluated in $O(|ub - lb|)$ steps. The size of domain $\text{dom}(S_i)$ therefore requires $O(|ub - lb||C|)$ steps to compute. Enumerating can also proceede level by level without taking into account the required elements lb since they belong to all sets in the domain. The first set on level k can be obtained from the lexicographic lower bound l. If $|l| \neq k$, we have to find the first set l' of cardinality k that is lexicographically greater than l. If $|l| < k$, we simply add to set l the $k - |l|$ smallest elements in $ub - lb - l$. Suppose $|l| > k$ and consider the binary representation of l. Let p be the k^{th} heaviest bit set to 1 in l. We add one to bit p and propagate carries and we set all bits before p to 0. We obtain a bit vector l' representing a set with no more than k elements. If $|l'| < k$, we add the first $k - |l'|$ elements in $ub - lb - l'$ to l' and obtain the first set of cardinality k.

Once the first set at level k has been computed, subsequent sets can be obtained using Algorithm 2. Obtaining the first set of each level costs $O(|\text{dom}(C)|$ $|ub - lb|)$ and cumulative calls to Algorithm 2 cost $O(\sum_{i \in \text{dom}(C)} i + |dom(S)|)$. Enumerating the domain therefore requires $O(|\text{dom}(C)||ub - lb| + |\text{dom}(S)|)$ steps.

5 All-Different on Tuples

A tuple t is an ordered sequence of n elements that allows multiple occurrences. Like sets, there are different ways to represent the domain of a tuple. The most common way is simply by associating an integer variable to each of the tuple components. A tuple of size n is therefore represented by n integer variables $x_1, \ldots, x_n$.

To apply an All-Different constraint to a set of tuples, a common solution is to create an integer variable t for each tuple. If each component x_i ranges from 0 to c_i exclusively, we add the following channeling constraint between tuple t and its components.

$$t = ((((x_1 c_2 + x_2)c_3 + x_3)c_4 + x_4)\ldots)c_n + x_n = \sum_i^n \left(x_i \prod_{j=i+1}^n c_j \right)$$

This technique suffers from either inefficient or ineffective channeling between variable t and the components x_i. Most constraint libraries enforce bound consistency on t. A modification to the domain of x_i does not affect t if the bounds of $\text{dom}(x_i)$ remain unchanged. Conversely, even if all tuples encoded in $\text{dom}(t)$ have $x_i \neq v$, value v will most often not be removed from $\text{dom}(x_i)$. On the other hand, enforcing domain consistency typically requires $O(n^k)$ steps where k is the size of the tuple.

To address this issue, one can define a tuple variable whose domain is defined by the domains of its components.

$$\text{dom}(t) = \text{dom}(x_1) \times \ldots \times \text{dom}(x_n)$$

The size of such a domain is given by the following equation which can be computed in $O(n)$ steps.

$$|\text{dom}(t)| = \prod_{i=1}^{n} |\text{dom}(x_i)|$$

The domain of a tuple variable can be enumerated using Algorithm 3. Assuming the domain of all component variables have the same size, Algorithm 3 runs in $O(|dom(t)|)$ which is optimal.

As Sadler and Gervet [7] did for sets, we can add lexicographical bounds to tuples in order to better express the values the domain contains. Let l and u be these lexicographical bounds.

$$\text{dom}(t) = \{t \mid t[i] \in \text{dom}(x_i), l \leq t \leq u\}$$

Let $idx(v, x)$ be the number of values smaller than v in the domain of the integer variable x. More formally, $idx(v, x) = |\{w \in \text{dom}(x) \mid w < v\}|$. Assuming $idx(v, x)$ has a running time complexity of $O(\log(|\text{dom}(x)|))$, the size of