Boi Faltings
Adrian Petcu
François Fages
Francesca Rossi (Eds.)

LNAI 3419

Recent Advances in Constraints

Joint ERCIM/CoLogNET International Workshop on Constraint Solving and Constraint Logic Programming, CSCLP 2004
Lausanne, Switzerland, June 2004
Revised Selected and Invited Papers

Springer

Boi Faltings Adrian Petcu
François Fages Francesca Rossi (Eds.)

Recent Advances in Constraints

Joint ERCIM/CoLogNet International Workshop
on Constraint Solving and Constraint Logic Programming, CSCLP 2004
Lausanne, Switzerland, June 23-25, 2004
Revised Selected and Invited Papers

Springer

Series Editors

Jaime G. Carbonell, Carnegie Mellon University, Pittsburgh, PA, USA
Jörg Siekmann, University of Saarland, Saarbrücken, Germany

Volume Editors

Boi Faltings
Adrian Petcu
Ecole Polytechnique Federale de Lausanne (EPFL)
School of Computer and Communication Sciences
Institute of Core Computing Science
Artificial Intelligence Laboratory
IN (Ecublens), 1015 Lausanne, Switzerland
E-mail: {boi.faltings,adrian.petcu}@epfl.ch

François Fages
Institut National de Recherche en Informatique et en Automatique - INRIA
Domaine de Voluceau, Rocquencourt, BP 105, 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: 2005921905

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

ISSN 0302-9743
ISBN 3-540-25176-6 Springer Berlin Heidelberg New York

Springer is a part of Springer Science+Business Media

springeronline.com

Printed in Germany

Typesetting: Camera-ready by author, data conversion by Olgun Computergrafik
Printed on acid-free paper SPIN: 11402763 06/3142 5 4 3 2 1 0

Lecture Notes in Artificial Intelligence 3419
Edited by J. G. Carbonell and J. Siekmann

Subseries of Lecture Notes in Computer Science

Preface

Constraint programming is a very successful fifth-generation software technology with a wide range of applications. It has attracted a large community of researchers that is particularly strong in Europe.

In particular, constraint programming is the focus of the Working Group on Constraints of the European Research Consortium for Informatics and Mathematics (ERCIM) as well as a major interest of the European Network on Computational Logic (CoLogNET). These groups jointly sponsored a workshop on Constraint Satisfaction and Constraint Logic Programming (CSCLP 2004) held June 23–25 at the Ecole Polytechnique Fédérale de Lausanne (EPFL) in Switzerland. It was hosted by the Artificial Intelligence Laboratory of the EPFL, which is also a member of both groups.

This book presents a collection of papers that are either revised and extended versions of papers accepted at the workshop, or were submitted in response to the open call for papers that followed. The 15 papers in this volume were selected from 30 submissions by rigorous peer review.

The editors would like to take the opportunity to thank all authors and reviewers for the hard work they contributed to producing this volume. We also thank ERCIM and CoLogNET for their support of the workshop and the field of constraint programming in general. We hope the reader will find this volume helpful for advancing their understanding of issues in constraint programming.

December 2004

Boi Faltings
Adrian Petcu
François Fages
Francesca Rossi

Organization

This workshop was jointly organized as the 9th Meeting of the ERCIM Working Group on Constraints, coordinated by François Fages, and the 2nd Annual Workshop of the CoLogNET area on Constraint Logic Programming, coordinated by Francesca Rossi.

Organizing Institutes

The organization was handled by the EPFL, INRIA and the University of Padua.

Organizing and Program Committee

Boi Faltings	EPFL, Switzerland
Adrian Petcu	EPFL, Switzerland
François Fages	INRIA, France
Francesca Rossi	University of Padua, Italy

Sponsoring Organizations

Referees

Ola Angelsmark
Alexis Anglada
Roman Bartak
Nicolas Beldiceanu
Claudio Bettini
Stefano Bistarelli
Lucas Bordeaux
Berthe Y. Choueiry
François Fages
Jie Fang
Stephan Frank
Renker Gerrit
Joel Gompert
Venkata Praveen Guddeti
Brahim Hnich
Petra Hofstedt
Alan Holland
Peter Jonsson
Irit Katriel
T.K. Satish Kumar
Arnaud Lallouet
Steve Prestwich
Paraskevi Raftopoulou
Arathi Ramani
Stefan Ratschan
Igor Razgon
Dirk Reckmann
Francesca Rossi
Olga Tveretina
Petr Vilim
Richard Wallace
Toby Walsh
Armin Wolf
Neil Yorke-Smith
Yaling Zheng
Peter Zoeteweij

Lecture Notes in Artificial Intelligence (LNAI)

Vol. 3452: F. Baader, A. Voronkov (Eds.), Logic for Programming, Artificial Intelligence, and Reasoning. XI, 562 pages. 2005.

Vol. 3419: B. Faltings, A. Petcu, F. Fages, F. Rossi (Eds.), Constraint Satisfaction and Constraint Logic Programming. X, 217 pages. 2005.

Vol. 3416: M. Böhlen, J. Gamper, W. Polasek, M.A. Wimmer (Eds.), E-Government: Towards Electronic Democracy. XIII, 311 pages. 2005.

Vol. 3415: P. Davidsson, B. Logan, K. Takadama (Eds.), Multi-Agent and Multi-Agent-Based Simulation. X, 265 pages. 2005.

Vol. 3403: B. Ganter, R. Godin (Eds.), Formal Concept Analysis. XI, 419 pages. 2005.

Vol. 3398: D.-K. Baik (Ed.), Systems Modeling and Simulation: Theory and Applications. XIV, 733 pages. 2005.

Vol. 3397: T.G. Kim (Ed.), Artificial Intelligence and Simulation. XV, 711 pages. 2005.

Vol. 3396: R.M. van Eijk, M.-P. Huget, F. Dignum (Eds.), Agent Communication. X, 261 pages. 2005.

Vol. 3394: D. Kudenko, D. Kazakov, E. Alonso (Eds.), Adaptive Agents and Multi-Agent Systems III. IIX, 313 pages. 2005.

Vol. 3374: D. Weyns, H.V.D. Parunak, F. Michel (Eds.), Environments for Multi-Agent Systems. X, 279 pages. 2005.

Vol. 3369: V.R. Benjamins, P. Casanovas, J. Breuker, A. Gangemi (Eds.), Law and the Semantic Web. XII, 249 pages. 2005.

Vol. 3366: I. Rahwan, P. Moraitis, C. Reed (Eds.), Argumentation in Multi-Agent Systems. XII, 263 pages. 2005.

Vol. 3359: G. Grieser, Y. Tanaka (Eds.), Intuitive Human Interfaces for Organizing and Accessing Intellectual Assets. XIV, 257 pages. 2005.

Vol. 3346: R.H. Bordini, M. Dastani, J. Dix, A.E.F. Seghrouchni (Eds.), Programming Multi-Agent Systems. XIV, 249 pages. 2005.

Vol. 3345: Y. Cai (Ed.), Ambient Intelligence for Scientific Discovery. XII, 311 pages. 2005.

Vol. 3343: C. Freksa, M. Knauff, B. Krieg-Brückner, B. Nebel, T. Barkowsky (Eds.), Spatial Cognition IV. Reasoning, Action, and Interaction. XIII, 519 pages. 2005.

Vol. 3339: G.I. Webb, X. Yu (Eds.), AI 2004: Advances in Artificial Intelligence. XXII, 1272 pages. 2004.

Vol. 3336: D. Karagiannis, U. Reimer (Eds.), Practical Aspects of Knowledge Management. X, 523 pages. 2004.

Vol. 3327: Y. Shi, W. Xu, Z. Chen (Eds.), Data Mining and Knowledge Management. XIII, 263 pages. 2005.

Vol. 3315: C. Lemaître, C.A. Reyes, J.A. González (Eds.), Advances in Artificial Intelligence – IBERAMIA 2004. XX, 987 pages. 2004.

Vol. 3303: J.A. López, E. Benfenati, W. Dubitzky (Eds.), Knowledge Exploration in Life Science Informatics. X, 249 pages. 2004.

Vol. 3276: D. Nardi, M. Riedmiller, C. Sammut, J. Santos-Victor (Eds.), RoboCup 2004: Robot Soccer World Cup VIII. XVIII, 678 pages. 2005.

Vol. 3275: P. Perner (Ed.), Advances in Data Mining. VIII, 173 pages. 2004.

Vol. 3265: R.E. Frederking, K.B. Taylor (Eds.), Machine Translation: From Real Users to Research. XI, 392 pages. 2004.

Vol. 3264: G. Paliouras, Y. Sakakibara (Eds.), Grammatical Inference: Algorithms and Applications. XI, 291 pages. 2004.

Vol. 3259: J. Dix, J. Leite (Eds.), Computational Logic in Multi-Agent Systems. XII, 251 pages. 2004.

Vol. 3257: E. Motta, N.R. Shadbolt, A. Stutt, N. Gibbins (Eds.), Engineering Knowledge in the Age of the Semantic Web. XVII, 517 pages. 2004.

Vol. 3249: B. Buchberger, J.A. Campbell (Eds.), Artificial Intelligence and Symbolic Computation. X, 285 pages. 2004.

Vol. 3248: K.-Y. Su, J. Tsujii, J.-H. Lee, O.Y. Kwong (Eds.), Natural Language Processing – IJCNLP 2004. XVIII, 817 pages. 2005.

Vol. 3245: E. Suzuki, S. Arikawa (Eds.), Discovery Science. XIV, 430 pages. 2004.

Vol. 3244: S. Ben-David, J. Case, A. Maruoka (Eds.), Algorithmic Learning Theory. XIV, 505 pages. 2004.

Vol. 3238: S. Biundo, T. Frühwirth, G. Palm (Eds.), KI 2004: Advances in Artificial Intelligence. XI, 467 pages. 2004.

Vol. 3230: J.L. Vicedo, P. Martínez-Barco, R. Muñoz, M. Saiz Noeda (Eds.), Advances in Natural Language Processing. XII, 488 pages. 2004.

Vol. 3229: J.J. Alferes, J. Leite (Eds.), Logics in Artificial Intelligence. XIV, 744 pages. 2004.

Vol. 3228: M.G. Hinchey, J.L. Rash, W.F. Truszkowski, C.A. Rouff (Eds.), Formal Approaches to Agent-Based Systems. VIII, 290 pages. 2004.

Vol. 3215: M.G.. Negoita, R.J. Howlett, L.C. Jain (Eds.), Knowledge-Based Intelligent Information and Engineering Systems, Part III. LVII, 906 pages. 2004.

Vol. 3214: M.G.. Negoita, R.J. Howlett, L.C. Jain (Eds.), Knowledge-Based Intelligent Information and Engineering Systems, Part II. LVIII, 1302 pages. 2004.

Vol. 3213: M.G.. Negoita, R.J. Howlett, L.C. Jain (Eds.), Knowledge-Based Intelligent Information and Engineering Systems, Part I. LVIII, 1280 pages. 2004.

Vol. 3209: B. Berendt, A. Hotho, D. Mladenic, M. van Someren, M. Spiliopoulou, G. Stumme (Eds.), Web Mining: From Web to Semantic Web. IX, 201 pages. 2004.

Vol. 3206: P. Sojka, I. Kopecek, K. Pala (Eds.), Text, Speech and Dialogue. XIII, 667 pages. 2004.

Vol. 3202: J.-F. Boulicaut, F. Esposito, F. Giannotti, D. Pedreschi (Eds.), Knowledge Discovery in Databases: PKDD 2004. XIX, 560 pages. 2004.

Vol. 3201: J.-F. Boulicaut, F. Esposito, F. Giannotti, D. Pedreschi (Eds.), Machine Learning: ECML 2004. XVIII, 580 pages. 2004.

Vol. 3194: R. Camacho, R. King, A. Srinivasan (Eds.), Inductive Logic Programming. XI, 361 pages. 2004.

Vol. 3192: C. Bussler, D. Fensel (Eds.), Artificial Intelligence: Methodology, Systems, and Applications. XIII, 522 pages. 2004.

Vol. 3191: M. Klusch, S. Ossowski, V. Kashyap, R. Unland (Eds.), Cooperative Information Agents VIII. XI, 303 pages. 2004.

Vol. 3187: G. Lindemann, J. Denzinger, I.J. Timm, R. Unland (Eds.), Multiagent System Technologies. XIII, 341 pages. 2004.

Vol. 3176: O. Bousquet, U. von Luxburg, G. Rätsch (Eds.), Advanced Lectures on Machine Learning. IX, 241 pages. 2004.

Vol. 3171: A.L.C. Bazzan, S. Labidi (Eds.), Advances in Artificial Intelligence – SBIA 2004. XVII, 548 pages. 2004.

Vol. 3159: U. Visser, Intelligent Information Integration for the Semantic Web. XIV, 150 pages. 2004.

Vol. 3157: C. Zhang, H. W. Guesgen, W.K. Yeap (Eds.), PRICAI 2004: Trends in Artificial Intelligence. XX, 1023 pages. 2004.

Vol. 3155: P. Funk, P.A. González Calero (Eds.), Advances in Case-Based Reasoning. XIII, 822 pages. 2004.

Vol. 3139: F. Iida, R. Pfeifer, L. Steels, Y. Kuniyoshi (Eds.), Embodied Artificial Intelligence. IX, 331 pages. 2004.

Vol. 3131: V. Torra, Y. Narukawa (Eds.), Modeling Decisions for Artificial Intelligence. XI, 327 pages. 2004.

Vol. 3127: K.E. Wolff, H.D. Pfeiffer, H.S. Delugach (Eds.), Conceptual Structures at Work. XI, 403 pages. 2004.

Vol. 3123: A. Belz, R. Evans, P. Piwek (Eds.), Natural Language Generation. X, 219 pages. 2004.

Vol. 3120: J. Shawe-Taylor, Y. Singer (Eds.), Learning Theory. X, 648 pages. 2004.

Vol. 3097: D. Basin, M. Rusinowitch (Eds.), Automated Reasoning. XII, 493 pages. 2004.

Vol. 3071: A. Omicini, P. Petta, J. Pitt (Eds.), Engineering Societies in the Agents World. XIII, 409 pages. 2004.

Vol. 3070: L. Rutkowski, J. Siekmann, R. Tadeusiewicz, L.A. Zadeh (Eds.), Artificial Intelligence and Soft Computing - ICAISC 2004. XXV, 1208 pages. 2004.

Vol. 3068: E. André, L. Dybkjær, W. Minker, P. Heisterkamp (Eds.), Affective Dialogue Systems. XII, 324 pages. 2004.

Vol. 3067: M. Dastani, J. Dix, A. El Fallah-Seghrouchni (Eds.), Programming Multi-Agent Systems. X, 221 pages. 2004.

Vol. 3066: S. Tsumoto, R. Słowiński, J. Komorowski, J.W. Grzymała-Busse (Eds.), Rough Sets and Current Trends in Computing. XX, 853 pages. 2004.

Vol. 3065: A. Lomuscio, D. Nute (Eds.), Deontic Logic in Computer Science. X, 275 pages. 2004.

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

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

Vol. 3055: H. Christiansen, M.-S. Hacid, T. Andreasen, H.L. Larsen (Eds.), Flexible Query Answering Systems. X, 500 pages. 2004.

Vol. 3048: P. Faratin, D.C. Parkes, J.A. Rodríguez-Aguilar, W.E. Walsh (Eds.), Agent-Mediated Electronic Commerce V. XI, 155 pages. 2004.

Vol. 3040: R. Conejo, M. Urretavizcaya, J.-L. Pérez-de-la-Cruz (Eds.), Current Topics in Artificial Intelligence. XIV, 689 pages. 2004.

Vol. 3035: M.A. Wimmer (Ed.), Knowledge Management in Electronic Government. XII, 326 pages. 2004.

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

Vol. 3030: P. Giorgini, B. Henderson-Sellers, M. Winikoff (Eds.), Agent-Oriented Information Systems. XIV, 207 pages. 2004.

Vol. 3029: B. Orchard, C. Yang, M. Ali (Eds.), Innovations in Applied Artificial Intelligence. XXI, 1272 pages. 2004.

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

Vol. 3020: D. Polani, B. Browning, A. Bonarini, K. Yoshida (Eds.), RoboCup 2003: Robot Soccer World Cup VII. XVI, 767 pages. 2004.

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

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

Vol. 2990: J. Leite, A. Omicini, L. Sterling, P. Torroni (Eds.), Declarative Agent Languages and Technologies. XII, 281 pages. 2004.

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

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

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

Vol. 2969: M. Nickles, M. Rovatsos, G. Weiss (Eds.), Agents and Computational Autonomy. X, 275 pages. 2004.

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

Table of Contents

Constraint Propagation

GCC-Like Restrictions on the *Same* Constraint 1
Nicolas Beldiceanu, Irit Katriel, and Sven Thiel

A Note on Bilattices and Open Constraint Programming 12
Arnaud Lallouet

Pruning by Equally Constrained Variables 26
Igor Razgon and Amnon Meisels

Search

Trying Again to Fail-First 41
J. Christopher Beck, Patrick Prosser, and Richard J. Wallace

Characterization of a New Restart Strategy
for Randomized Backtrack Search 56
Venkata Praveen Guddeti and Berthe Y. Choueiry

Dynamic Distributed BackJumping 71
Viet Nguyen, Djamila Sam-Haroud, and Boi Faltings

A Value Ordering Heuristic for Local Search
in Distributed Resource Allocation 86
Adrian Petcu and Boi Faltings

Automatically Exploiting Symmetries in Constraint Programming 98
Arathi Ramani and Igor L. Markov

New Structural Decomposition Techniques
for Constraint Satisfaction Problems 113
Yaling Zheng and Berthe Y. Choueiry

Applications

Algorithms for the Maximum Hamming Distance Problem 128
Ola Angelsmark and Johan Thapper

A System Prototype for Solving Multi-granularity Temporal CSP 142
Claudio Bettini, Sergio Mascetti, and Vincenzo Pupillo

Computing Equilibria Using Interval Constraints 157
Lucas Bordeaux and Brice Pajot

Constraint-Based Approaches to the Covering Test Problem 172
Brahim Hnich, Steven Prestwich, and Evgeny Selensky

Super Solutions for Combinatorial Auctions 187
Alan Holland and Barry O'Sullivan

Better Propagation
for Non-preemptive Single-Resource Constraint Problems 201
Armin Wolf

Author Index 217

GCC-Like Restrictions on the *Same* Constraint

Nicolas Beldiceanu[1], Irit Katriel[2], and Sven Thiel[2]

[1] LINA FRE CNRS 2729, École des Mines de Nantes, FR-44307 Nantes Cedex 3, France
Nicolas.Beldiceanu@emn.fr

[2] Max-Planck-Institut für Informatik, Stuhlsatzenhausweg 85, 66123 Saarbrücken, Germany
{irit,sthiel}@mpi-sb.mpg.de

Abstract. The *Same* constraint takes two sets of variables X and Z such that $|X| = |Z|$ and assigns values to them such that the multiset of values assigned to the variables in X is equal to the multiset of values assigned to the variables in Z. In this paper we extend the *Same* constraint in a GCC-like manner by adding cardinality requirements on the values. That is, for each value we have a lower and upper bound on the number of variables that can be assigned this value. We show an algorithm that achieves arc-consistency for this constraint and a faster algorithm that achieves bound-consistency for a restricted case of it.

1 Introduction

The *Same*$(X = \{x_1,\dots,x_n\}, Z = \{z_1,\dots,z_n\})$ constraint [2] is defined on two sets X and Z of distinct variables such that $|X| = |Z|$ and each $a \in X \cup Z$ has a finite domain $D(a)$. A solution is an assignment of values to the variables such that the value assigned to each variable belongs to its domain and the multiset of values assigned to the variables of X is identical to the multiset of values assigned to the variables of Z.

This constraint can be used to model simple scheduling problems such as the one described in [2]: The organization Doctors Without Borders [11] has a list of doctors and a list of nurses, each of whom volunteered to go on one rescue mission in the next year. Each volunteer specifies a list of possible dates and each mission should include one doctor and one nurse. The task is to produce a list of pairs such that each pair includes a doctor and a nurse who are available on the same date and each volunteer appears in exactly one pair.

In the setting described above, the number of potential rescue missions on each day is infinite, so we do not care how the doctor-nurse pairs are distributed between the dates. This paper deals with a variant of *Same* which we call *Same With Cardinalities* (*SWC*) and which allows us to model the doctor-nurse problem when for each date there is a minimum number of missions that must be staffed and a maximum number of missions that are possible. The reader should be reminded of the Global Cardinality Constraint (*GCC*) [5, 7, 8, 10], which is defined on one set of variables and specifies for each value the minimum and maximum number of variables that are to be assigned this value.

Formally, the *SWC*$(X = \{x_1,\dots,x_n\}, Z = \{z_1,\dots,z_n\}, C = \{c_{v_1},\dots,c_{v_{n'}}\})$ constraint is specified on two sets X and Z, each containing n assignment variables, and a third set C of n' count variables. With each assignment variable $a \in X \cup Z$ we associate a domain

B. Faltings et al. (Eds.): CSCLP 2004, LNAI 3419, pp. 1–11, 2005.

$D(a) \subseteq D = \{v_1, \ldots, v_{n'}\}$. The count variable c_{v_i} refers to $v_i \in D$ and its domain is an interval $D(c_{v_i}) = [L_i, U_i]$. A solution to the *SWC* constraint is an assignment of values to the variables in $X \cup Z$ such that:

- Each $a \in X \cup Z$ is assigned a value in its domain $D(a)$.
- Each c_{v_i} is assigned a value in the interval $D(c_{v_i})$.
- The multiset of values assigned to the variables of X is equal to the multiset of values assigned to the variables of Z.
- The number of variables in X (and hence also in Z) which are assigned the value v_i is equal to the number assigned to c_{v_i}.

In other words, for a tuple $t \in D^n$ and a value $v \in D$, let $occ(v,t)$ be the number of occurrences of the value v in t. Then the set S of all solutions to the constraint is:

$$\begin{aligned} S = \{(u_1, \ldots, u_n \,;\, w_1, \ldots, w_n \,;\, o_1, \ldots, o_{n'}) \mid \\ \forall_j \; u_j \in D(x_j) \;\wedge\; \forall_j \; w_j \in D(z_j) \;\wedge \\ \forall_i \; occ(v_i, (u_1, \ldots, u_n)) = occ(v_i, (w_1, \ldots, w_n)) = o_i \in D(c_{v_i})\}. \end{aligned}$$

1.1 Arc-Consistency and Bound-Consistency

Given a constraint with domains for the variables, the first question is whether $S \neq \emptyset$, which means that there is at least one assignment of values to the variables that satisfies the constraint. The second question is whether there are efficient *filtering* algorithms for this constraint. That is, algorithms that shrink the domains of the variables by removing values that cannot participate in any solution. The *arc-consistency* problem is to reduce the domain of each variable a such that $D(a)$ is the projection of S onto the component that corresponds to a. That is, a value v remains in $D(a)$ iff there is a solution to the constraint in which a is assigned the value v. In the *bound-consistency* problem we assume that the values are linearly arranged, i.e., $v_1 < \cdots < v_{n'}$ and for each $a \in X \cup Z$, $D(a)$ is a contiguous interval of values, i.e., $D(a) = [\underline{D}(a), \overline{D}(a)]$. The problem is to shrink the intervals to the minimum sizes such that $S \subseteq D(x_1) \times \cdots \times D(x_n) \times D(z_1) \times \cdots \times D(z_n) \times D(c_{v_1}) \times \cdots \times D(c_{v_{n'}})$. I.e., the domain of the kth variable is bound-consistent iff S contains at least one tuple whose kth component equals the smallest (largest) value in it.

1.2 $SWC = 2 \times GCC$?

The *SWC* constraint can be modeled by two *Global Cardinality* constraints [5, 7, 8, 10], one on the set X and the other on the set Z, where count variables which are associated with the same value are not duplicated. We show here that consistency for all of the variables of the *GCC* constraints (including assignment and count variables) does not imply consistency for the *SWC* constraint.

In our example, $|X| = |Z| = 2$ and $|Y| = 4$. The domains of the assignment variables are: $D(x_1) = \{1,2\}$, $D(x_2) = \{3,4\}$, $D(z_1) = \{1,2,3,4\}$ and $D(z_2) = \{3,4\}$ and the domain of each count variable is $\{0,1\}$. By examining the variable-value graphs[1] shown in Figure 1, one can easily see that all values are consistent with respect to the two *GCC*

[1] This construction will be formally defined in Section 2.

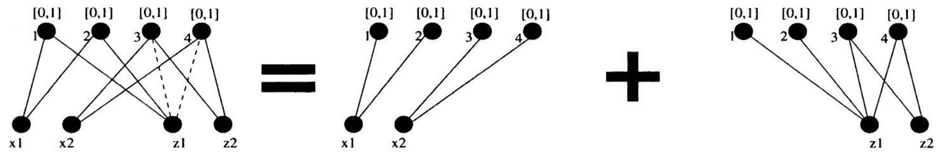

Fig. 1. Example showing that consistency of the two *GCC*'s does not imply consistency of the *SWC* constraint (even when all cardinalities are in [0,1]).

constraints $GCC(\{x_1,x_2\},\{c_{v_1},c_{v_2},c_{v_3},c_{v_4}\})$ and $GCC(\{z_1,z_2\},\{c_{v_1},c_{v_2},c_{v_3},c_{v_4}\})$, but that an arc-consistency or bound-consistency computation for the *SWC* constraint would remove 3 and 4 from the domain of z_1: If z_1 is assigned 3 or 4, then 1 and 2 cannot both be assigned to the same number of variables from X and Z because one of them must be assigned to x_1 and neither can be assigned to z_2.

1.3 Filtering with Flows/Matchings

Network flows were used to design filtering algorithms for several globals constraints. These algorithms follow a general scheme: the constraint is modeled as a network such that there is a correspondence between feasible integral flows in the network and solutions to the constraint. The algorithm finds a feasible flow in this network, constructs the residual graph with respect to this flow and computes the strongly connected components (SCCs) of the residual graph. Then, it is shown how to use the flow and the SCCs to reduce the domains of the variables to arc-consistency or bound-consistency.

Régin was the first to use this approach when he designed an arc-consistency algorithm for the *AllDifferent* constraint [9], which he later generalized for the Global Cardinality Constraint [10]. Mehlhorn and Thiel [6] showed that this scheme gives rise to a faster bound-consistency algorithm for *AllDifferent*. They noticed that in the bound-consistency problem, the network on which the flow and SCC computations need to be performed has a certain structure, *convexity*, which can be exploited in order to perform these computations more efficiently. Katriel and Thiel [5] showed how to exploit convexity to achieve a fast bound-consistency algorithm for *GCC*. Later, the authors [2] defined the *Same* and *UsedBy* constraints and designed arc-consistency and bound-consistency algorithms for them, which also follow the flow-based paradigm. The networks that model the *Same* and *UsedBy* constraints are more complex than the ones used for *AllDifferent* and *GCC*, a fact that also complicates the filtering algorithms, in particular the efficient bound-consistency algorithms. In this paper we show how to model the *SWC* constraint. The network we use resembles the one that was used for the *Same* constraint, but the capacity requirements for the values add a new twist: until now, all networks consisted of a bipartite graph with a node for each value on one side and a node for each variable on the other side, plus two special nodes. The network we use for *SWC* breaks away from this line: each value is modeled by two nodes that are connected by an edge. This structure complicates things even further, in particular in the bound-consistency computation.

1.4 Filtering for the *SWC* Constraint

In the next section we present filtering algorithms for the assignment variables of the *SWC* constraint. The first algorithm achieves arc-consistency and runs in time $O(n^2n')$.

The second algorithm achieves bound-consistency in the restricted case of *SWC* in which $D(c_{v_i}) = [0,1]$ for all $1 \leq i \leq n'$. It runs in time $O(nn')$. As we have noted above, *SWC* is a *GCC*-like restriction of the *Same* constraint. Similarly, the case in which $D(c_{v_i}) = [0,1]$ for all i is analogous to the *AllDifferent* constraint [6, 9, 12] which is the special case of *GCC* in which the capacities for all variables are $[0,1]$.

2 An Arc-Consistency Algorithm for *SWC*

We represent the *SWC* constraint as a flow problem in a directed graph $\vec{G} = (V, \vec{E})$ which we call the *variable-value graph*. The nodes of $\vec{G}$ are $V = \{X \cup Z \cup Y_{in} \cup Y_{out} \cup \{s,t\}\}$ where $Y_{in} = \{y_1^{in}, \ldots y_{n'}^{in}\}$ and $Y_{out} = \{y_1^{out}, \ldots y_{n'}^{out}\}$. In other words, there is a node a for each variable $a \in X \cup Z$, there are two nodes y_i^{in}, y_i^{out} for each value v_i where $1 \leq i \leq n'$ and there are two additional nodes s and t. The edges of $\vec{G}$ are:

- For each $x_j \in X$ and $i \in D(x_j)$, $(x_j, y_i^{in}) \in \vec{E}$ with capacities $[0,1]$.
- For each $z_j \in Z$ and $i \in D(z_j)$, $(y_i^{out}, z_j) \in \vec{E}$ with capacities $[0,1]$.
- For each $1 \leq i \leq n'$, $(y_i^{in}, y_i^{out}) \in \vec{E}$ with capacities $[L_i, U_i]$.
- For each $1 \leq j \leq n$, $(s, x_j) \in \vec{E}$ with capacities $[1,1]$.
- For each $1 \leq j \leq n$, $(z_j, t) \in \vec{E}$ with capacities $[1,1]$.
- $(t,s) \in \vec{E}$ with capacities $[n,n]$.

Table 1. Domains of the variables for our example.

j	$D(x_j)$	$D(z_j)$
1	[1,2]	[2,3]
2	[3,4]	[4,5]
3	[4,6]	[4,5]

Table 2. Domains of the count variables for our example.

i	1	2	3	4	5	6
$[L_i, U_i]$	[0,1]	[1,2]	[0,3]	[1,4]	[0,2]	[0,1]

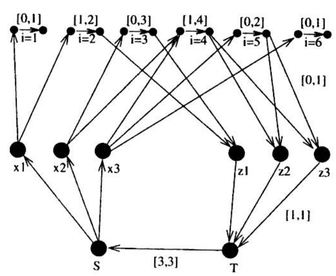

Fig. 2. The variable-value graph for the example in Tables 1 and 2.

Figure 2 shows the graph $\vec{G}$ for the following input. $|X| = |Z| = 3$ and $|Y| = 6$. The domains of the variables of $X \cup Z$ are as in Table 1 and the domains of the count variables are as in Table 2.

The following definition comes from flow theory. See Figure 3 for an example of a feasible flow.

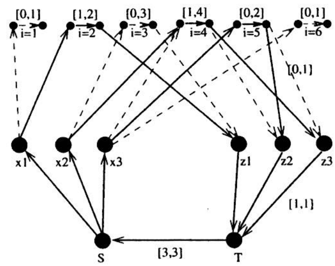

Fig. 3. An integral feasible flow in the graph of Figure 2. The solid edges carry flow and the dashed edges do not.

Definition 1. *Given a directed graph $\vec{G} = (V, \vec{E})$ with lower and upper capacities l_e, u_e for each edge $e \in \vec{E}$, a* feasible flow *in $\vec{G}$ is a function $f : E \longrightarrow \mathbb{R}$ such that*

1. **Flow Conservation:** *For each node $v \in V$,*

$$\sum_{\{u|(v,u)\in\vec{E}\}} f(v,u) = \sum_{\{w|(w,v)\in\vec{E}\}} f(w,v).$$

2. **Capacities:** *For each $e \in \vec{E}$, $l_e \leq f(e) \leq u_e$.*

An integral *feasible flow is a feasible flow such that for all $e \in \vec{E}$, $f(e)$ is an integer.*

Lemma 1. *There is a one-to-one correspondence between the integral feasible flows in $\vec{G}$ and the solutions to the constraint.*

Proof. Let f be an integral feasible flow in $\vec{G}$. For each $x \in X$, the amount of flow coming into x (from s) is exactly 1, hence there is exactly one $y \in Y_{in}$ which is connected to x by an edge that carries non-zero flow. Similarly, the flow out of each $z \in Z$ (to t) is exactly 1 so there is exactly one $y \in Y_{out}$ which is connected to z by an edge that carries non-zero flow. For each $a \in X \cup Z$, let $I(a)$ be the index of this node y. That is, $I(a) = i$ such that $y \in \{y_i^{in}, y_i^{out}\}$.

Then we can construct the solution

$$SWC(\{I(x_1), \ldots, I(x_n)\}, \{I(z_1), \ldots, I(z_n)\}, \{f(y_1^{in}, y_1^{out}), \ldots, f(y_{n'}^{in}, y_{n'}^{out})\}).$$

For all $a \in X \cup Z$, $I(a)$ is well defined. Since the edges $(x, y_{I(x)}^{in})$ and $(y_{I(z)}^{out}, z)$ carry flow, they exist in $\vec{G}$, which implies that $I(a) \in D(a)$ for all $a \in X \cup Z$. In addition, by flow conservation and by the choice of capacities for the edges (y_i^{in}, y_i^{out}) we have $L_i \leq |\{x \in X | I(x) = i\}| = |\{z \in Z | I(z) = i\}| \leq U_i$ for all $1 \leq i \leq n'$, so each value is assigned the same number of times to variables of X and Z, and this number is within its capacity requirements. Hence, the constraint is satisfied.

On the other hand, any solution $SWC(\{I(x_1), \ldots, I(x_n)\}, \{I(z_1), \ldots, I(z_n)\}, \{o_1, \ldots, o_{n'}\})$ where $I(a)$ is the value assigned to the variable a, allows us to construct an integral feasible flow f as follows.

- For each $x \in X$, $f(x, y_{I(x)}^{in}) = 1$ and $f(x, y_j^{in}) = 0$ for all $j \in D(x) \setminus I(x)$.
- For each $z \in Z$, $f(y_{I(z)}^{out}, z) = 1$ and $f(y_j^{out}, z) = 0$ for all $j \in D(z) \setminus I(z)$.

- For each $1 \leq i \leq n'$, $f(y_i^{in}, y_i^{out}) = o_i$.
- For each $x \in X$, $f(s,x) = 1$.
- For each $z \in Z$, $f(z,t) = 1$.
- $f(t,s) = n$.

Since $I(a) \in D(a)$ for all a, all the edges through which we wish to pass positive flow exist in the graph. In addition, since $I(a)$ is determined for all variables, we have that $f(t,s)$ is n and $f(s,x) = f(z,t) = 1$ for $x \in X$ or $z \in Z$. Since $L_i \leq |\{x \in X | I(x) = i\}| = o_i = |\{z \in Z | I(z) = i\}| \leq U_i$ for all $1 \leq i \leq n'$, we get that the total amount of flow into y_i^{in} is equal to the total amount of flow out of y_i^{out} and to the amount of flow through (y_i^{in}, y_i^{out}), and that it is within the capacity range of the edge (y_i^{in}, y_i^{out}). Hence the flow is an integral feasible flow. □

After finding an integral feasible flow f in $\vec{G}$, we construct the residual graph $\vec{G}_f = (V, \vec{E}_f)$. The edges in $\vec{E}_f$ are as follows. An edge between $a \in X \cup Z$ and $y \in Y_{in} \cup Y_{out}$ appears in $\vec{E}_f$ in its original orientation iff it carries flow zero and in its reverse direction iff it carries flow 1. The edge (y_i^{in}, y_i^{out}) exists iff $f(y_i^{in}, y_i^{out}) < U_i$ and the edge (y_i^{out}, y_i^{in}) exists iff $f(y_i^{in}, y_i^{out}) > L_i$. There are no edges touching s and t. Figure 4 shows the residual graph for our example, with respect to the flow of Figure 3. The following lemma states that we can use the residual graph to determine which edges of the graph are consistent.

Lemma 2. *Let $e = (u,v)$ be any edge in $\vec{G}_f$ with $u \in X \cup Z$ or $v \in X \cup Z$. Then $f'(e) = f(e)$ for all feasible flows f' in $\vec{G}$ iff u and v do not belong to the same strongly connected component (SCC) of $\vec{G}_f$.*

Proof. Standard flow theory. □

Lemma 3. *An edge $e = (u,v) \in \vec{G}_f$ with $u \in X \cup Z$ or $v \in X \cup Z$ is consistent iff $f(e) = 1$ or u and v belong to the same SCC.*

Proof. If $f(e) = 1$ then e participates in the solution that corresponds to the flow f and is therefore consistent. Otherwise, by Lemma 2 we get that there is a flow f' such that $f'(e) = 1$ (and hence a solution that uses the assignment represented by e) iff u and v belong to the same SCC. □

Lemma 3 implies the last step of the filtering algorithm: For each variable $a \in X$, remove a value i from $D(a)$ if $f(a, y_i^{in}) = 0$ and a, y_i^{in} do not belong to the same SCC

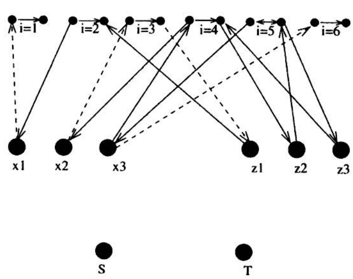

Fig. 4. The residual graph with respect to the flow shown in Figure 3. The dashed edges are not consistent.