CSCE 421/821: Foundations of Constraint Processing, Fall 2014

List of Projects

Berthe Y. Choueiry
Constraint Systems Laboratory
Department of Computer Science and Engineering
University of Nebraska-Lincoln
`choueiry@cse.unl.edu`
October 17, 2014

Distributed: Friday, October 17, 2014

Project selection: Wednesday, October 29, 2014 (by `handin`)

Progress reports: Wednesday, November 19, 2014 (by `handin`)

Final reports: Friday, December 5, 2014 (paper and by `handin`)

Presentations: M/W/F, December 8/10/12, 2014 (evening sessions scheduled as necessary). Also Wednesday, December 17, 2014 from 7:30am–9:30am.

Code and slides: Friday, December 12, 2014 (by `handin`)

Note: *Clearly acknowledge help received from anyone*. Always acknowledge your sources (URLs, books, class-notes, etc.).

Contents

---

1 Guidelines

Below is a non-exhaustive list of possible topics for semester projects. *If you have an idea for a project, do not hesitate to discuss it with the instructor.* There are main categories to choose from:

1. Implement and evaluate an algorithm, Sections 2.

2. Model and solve a problem as a CSP, Section 3.

3. Investigate an advanced theoretical concept, Section 4.

4. Conduct a critical literature survey of an advanced topic, Section 5.

Finally, a few suggestions for paper summaries and presentations to improve your grade are provided in Section 6.

More guidelines:

- Some projects may have enough substance to be conducted in a team of two students. When this is the case, *each student will have* to provide the instructor with an evaluation of the performance of his/her team partners. This feedback could be provided orally or by filling a standard form (ask instructor for the form).

- The same project may be chosen by more than one person or team. So, do not rush to 'reserve' yourself a project. If a project is selected by more than one person or team, we will carry out a comparison of the approach and results.

- Again, you are encouraged to design your own project proposal and discuss it with the instructor. Check the buzzwords appearing on the slides of CSP 101 (Overview), Part I and Part II and we can find papers on the topic for presentation, implementation, and/or classroom discussion in this semester and also in CSCE 921.

- Most of the papers of the last CP conferences are all available online. Feel free to check them:
  `http://www.informatik.uni-trier.de/~ley/db/conf/cp/cp2007.html`
  `http://www.informatik.uni-trier.de/~ley/db/conf/cp/cp2008.html`
  `http://www.informatik.uni-trier.de/~ley/db/conf/cp/cp2009.html`
  `http://www.informatik.uni-trier.de/~ley/db/conf/cp/cp2010.html`
  `http://www.informatik.uni-trier.de/~ley/db/conf/cp/cp2011.html`
  `http://www.informatik.uni-trier.de/~ley/db/conf/cp/cp2012.html`
  `http://www.informatik.uni-trier.de/~ley/db/conf/cp/cp2013.html`
  `http://www.informatik.uni-trier.de/~ley/db/conf/cp/cp2014.html`
  Papers are divided into: invited talks, application papers, full research papers, and short research papers. Quickly identify titles of interest to you, browse through the papers to see if they appeal to you. Come and discuss the marked papers with me and we can decide whether it is appropriate for implementation and testing in a project, paper presentation or paper summary.

  Further, most recent CP papers are now available online from Springer via the UNL library site `http://iris.unl.edu`. From an off-campus location, you still have to go through the same site but perhaps provide some UNL authentication/identification.

- Collaborate with a research assistant. Contact a research assistant and convince him/her to invest in you and work with you. Define with him/her a project to work on. He/she will provide mentoring and supervision in tight collaboration with the instructor. Current research assistants are: Daniel Geschwender, Anthony Schneider, Nathan Stender, and Robert Woodward.

---

2 Experimental evaluation of advanced algorithms

Every implementation should be tested, as applicable, on a real-world problem, randomly generated problems, and/or benchmark problems. Usually, results should be reported in terms of nodes visited, constraint checks, CPU time, and other applicable criteria. Generators for random CSPs are available. Tests should be conducted for various values of constraint tightness and density, and results should be averaged for at least 50 problem instances per measurement point. Details of the testing and evaluation methodologies should be discussed with the instructor on a case-by-case basis.

1. *Search strategy.* Study, implement, and test the search strategy known as *depth-bounded discrepancy search* of [Walsh 1997].

2. *Visualization of search.* Choose any of the projects below, simplify it in agreement with the instructor, implement it in Python, and generate a visualization of its operation using Sage.[1]

3. *Ordering heuristic.* Study, implement, and evaluate the ordering heuristics techniques proposed by Michel and Hentenryck [2012],

4. *Ordering heuristic.* Study, implement, and evaluate the ordering heuristics techniques proposed by Refalo [2004].

5. *Ordering heuristic.* Study, implement, and evaluate the ordering heuristics techniques proposed by Zanarini and Pesant [2007] and their application to `alldifferent` and `regular` constraints.

6. *Distributed CSPs.* Study, implement, and evaluate an algorithm for asynchronous backtracking such as the one by Zivan et al. [2007], Zivan and Meisels [2005], or by Maestre and Bessière [2004].

7. *Propagation algorithms: GAC.* Implement and evaluate the performance of solving non-binary CSPs with GAC2001 [Bessière *et al.* 2005].
   Comment: Suited for an undergraduate student.

8. *Parameterized lookahead.* Implement and compare the performance of FC, MAC, and p-MAC for solving binary CSPs. p-MAC is a lookahead algorithm proposed by a student of CSCE821 during Spring 2008 for controlling the depth of the lookahead according to a depth parameter p. The student will have to implement AC2001 as a basis for the lookahead. Please discuss with instructor if interested.

9. *Propagation algorithms: path consistency.* Implement and compare the performance of the following algorithms for path consistency: `PC-2` [Mackworth 1977], `PC-8` [Chmeiss and Jégou 1998], and `PC-2001` [Bessière *et al.* 2005]. No search.
   Comment: Suited for an undergraduate student.

10. *Propagation algorithms: path consistency.* Implement and compare the performance of the following algorithms for path consistency: `PC-2` [Mackworth 1977], `DPC` [Dechter 2003a], and `PPC` [Bliek and Sam-Haroud 1999].

11. *Propagation algorithms: path consistency.* Study, implement, and evaluate the algorithms for Path Consistency by Dual Consistency proposed by Lecoutre et al. [2007a].

12. *Comparing arc and path consistency in lookahead.* Implement and compare the performance of the following backtrack search algorithms:

[1] `http://www.sagemath.org/`

(a) Backtrack search with *dynamic* variable ordering and real full lookahead schema with AC.

(b) Backtrack search with *staticc* variable ordering and real full lookahead with DPC (ref. Section 4.2.2 and Figure 4.9 of your textbook [Dechter 2003b]. The variables are ordered using the maximum cardinality ordering on the triangulated graph of the binary CSP.

(c) Backtrack search with *dynamic* variable ordering and real full lookahead using `PC-8` [Chmeiss and Jégou 1998].

Comment: Suited for a team of two students. A student can choose to do a portion of the project.

13. *Propagation algorithms for the Latin Square.* Implement and compare the performance and pruning power of the following algorithms for solving the Quasigroup Completion Problem: AC, SAC, GAC, and SGAC. The student would be able to use and improve our current implementation for the Sudoku puzzle in Java.
Comment: Suited for an undergraduate student.

14. *Propagation algorithms for Sudoku.* Implement a more aggressive version of SGAC, S^2GAC, and see if it can solve, w/o search, the remaining 7 instances of the Sudoku puzzle. Compare S^2GAC to what dual SGAC would be [2007a]. The student would be able to use and improve our current implementation for the Sudoku puzzle in Java.
Comment: Suited for an undergraduate student.

15. *Propagation algorithms for the Kakuro.* Implement and compare the performance and pruning power of the propagation algorithms for solving the Kakuro puzzle [Simonis 2008; Cambazard]. The student may want to use and improve our current implementation for the Sudoku puzzle in Java.

16. *Propagation algorithms: Neighborhood Inverse Consistency*. Implement the propagation algorithm NIC [Freuder and Elfe 1996] on a binary CSP and RNIC (ref. Robert Woodward) on the dual of a binary CSP. Compare the pruning power of the two algorithms.
Comment: Suited for an undergraduate student.
For additional challenge, integrate the two algorithms in your backtrack search as full lookahead strategies and compare the pruning power of the two resulting algorithms.

17. *Propagation algorithms: global constraints*. Study, implement, and evaluate the propagation algorithm for the 'deviation' global constraint proposed by Schaus *et al.* [2007].

18. *Propagation algorithms: global constraints*. Study, implement, and evaluate the propagation algorithm for the global cardinality constraint proposed by Régin [1996].

19. *Propagation algorithms: Temporal Reasoning with qualitative constraints.* Study, implement, and test the algorithms for computing the minimal network on point algebra constraints proposed by Gerevini and Saetti [2007].

20. *Propagation algorithms: subgraph isomorphism.* Study, implement, and test the algorithms for filtering subgraph isomorphism proposed by Zampelli et al. [2007]. (Optional: Compare with a technique for the same purpose studied by the instructor.)

21. *Temporal Reasoning with metric constraints.* Study, implement, and test the algorithms for solving the DTP proposed by Kumar [2005].
Comment: Suited for a graduate student.

22. *Temporal Reasoning with metric constraints: Search.* Study, implement, and evaluate the backtrack search for solving the Disjunctive Temporal Problem (DTP) proposed in Tsamardinos and Pollack [2003].

23. *Symmetry.* Study, implement, and test the techniques proposed by Meseguer and Torras [2001] for exploiting symmetries in backtrack search.

24. *Symmetry.* Study, implement, and test the techniques proposed by Law et al. [2007] for breaking symmetries of interchangeable variables and values.

25. *Symmetry.* Study and implement the symmetry technique for syntactic symmetry of [Benhamou 1994]. Implement it in backtrack search and evaluate its performance and benefits. For increasing the challenge: implement it also for non-binary CSPs.

26. *Dominance.* Study, implement, and test the techniques proposed by Lecoutre et al. [2007b] for improving backtrack search by detecting dominance and pruning the search tree.

27. *Bundling strategies.* (Initial code exists in Common Lisp.) Evaluate bundling strategies on benchmark problems. Modify your FC search algorithm to exploit dynamic neighborhood interchangeability [Choueiry and Davis 2002]. Compare the original and modified algorithms.
    Comment: Suited for an undergraduate student.

28. *Backjumping on QCSPs.* Study, implement, and evaluate the algorithm backjumping for Quantified CSPs proposed by Bacchus and Stergiou [2007].

29. *Decomposition.* Study, implement, and evaluate the algorithm for decomposing CSPs of [Chmeiss *et al.* 2003].

3 Modeling

1. *CLUE as a CSP.* Design a "*Player X*" agent for the game of Clue. *Player X* has one SAT/CSP model for its own game and another for each other player in the game. It operates by finding the most informative query for itself and the least informative answer for its opponents. Build an interface that simulates the game and implements *Player X*. At this stage, the sequence of situations in a given game and the planning aspects should be ignored and replaced by a random generation of "question/answer" scenarios for *Player X*.
   Comment: An interesting instance of a distributed CSP with privacy constraints and redundant modeling. May become the basis of an MS thesis, perhaps more. For the semester project, you should ignore the planning-task component of the game. Highly challenging, highly creative.

2. *Modeling Software Engineering tasks as CSPs.* Either study the modeling of Program Verification as a CSP proposed by Collaviza and Rueher [2007], or propose a model of this or another task in Software Engineering as a CSP. Implement, test, and evaluate the proposed techniques on toy and/or benchmark problems.
   Comment: Instructor is not familiar with work in this area.

3. *Global constraints.* Research a number of global constraints, study their semantics, and investigate, implement, and evaluate the specialized propagation algorithms proposed in the literature for this purpose. You may refer to the catalogue in `http://www.emn.fr/x-info/sdemasse/gccat/` or proceedings of main conferences.

4 Research

1. *Learning with Conflict Directed Backjumping.* CBJ could benefit from maintaining the learned no-goods to improve pruning during search. Modify FC-CBJ to record and manage no-goods and exploit them during search.

2. *Theoretical aspects of Constraint Satisfaction.* Study the work of Atserias *et al.* [2007], Kolaitis and Vardi [2008], Feder and Vardi.[2] Alternatively, study the contributions in the Dagstuhl Seminar 09441 "The Constraint Satisfaction Problem: Complexity and Approximability" [Creignou *et al.* 2008].[3] You may choose to focus on the work of a particular author or any subset of them.
   Comment: Suited only for a student with a solid preparation in TCS.

3. *Generation of random solutions.* Study, implement and test the method for generating solutions to a CSP uniformly at random of [Dechter *et al.* 2002].

4. *Disjunctive decomposition.* Study the following disjunctive decomposition strategy: Each value for each variable is evaluated to assess whether its removal would cause a binary CSP to become disconnected. The technique is best used neighborhood interchangeability. Study the structure of the disconnected CSPs. For one of them, either solve it with search or repeat the operation. Focus on binary CSPs as a first approximation. Discuss topic with instructor.

5 Literature review

Conduct a critical review and a synthesis of an area where constraints are studied or applied, such as:

1. Tractability conditions and the broken triangle property, as documented in the recent papers by Martin Cooper. (Ask instructor for references.)

2. Multi-valued decision diagrams (MDD) are compact structures for storing partial solutions. Research, study, and compare how MDD are exploited in CP using the paper [Hoda *et al.* 2010].

3. Study of propagation in SAT solvers and comparison to consistency algorithm in CSP solvers.

4. Symmetric CSPs.

5. Hypertree decomposition (tractability studies of CSPs), e.g. [Hubie Chen 2005; Grohe and Marx 2006].

6. Use of SAT and constraints in Model Checking.

7. Soft constraints and preferences in CSPs: CP-nets.

8. Temporal reasoning.

9. Distributed CSPs.

10. Numeric (a.k.a. continuous) CSPs.

11. Relationship between CSPs and belief networks (recent work by Dechter).

[2] `www.cs.rice.edu/~vardi/papers/pods00t.ps.gz`
[3] `http://drops.dagstuhl.de/opus/frontdoor.php?source_opus=2371`

6 Some papers for summaries or presentations

Here is a list of papers you may want to study for a summary:

- Value ordering heuristic for finding all the solutions of a CSP [Smith and Sturdy 1997].
- CP toolboxes: Visualization Platform for CP [Simonis *et al.* 2010], Modeling, Debugging, and Visualization Environment [Bauer *et al.* 2010].
- Various papers on industrial applications of CP in telecommunications, planning and scheduling in a large oil pipeline network, protein structure prediction, etc. from the Proceedings of CP 2007, 2008, 2009, and 2010. Examples: Resource management [Asaf *et al.* 2010; van der Krogt *et al.* 2010; Lesaint *et al.* 2010], velocity tuning in UAVs [Soulignac *et al.* 2010], air traffic management [Blomdahl *et al.* 2010].
- Restart strategies for backtrack search [Wu and van Beek 2007]. Could be also used for a project: implement, test, and compared against the one in [Guddeti and Choueiry 2005].
- Tractability of perfect constraints [Salamon and Jeavons 2008].
- Semiring and soft constraints for diagnosis [Sachenbacher and Williams 2004].
- The paper on variable ordering heuristics by Beck et al. [2003]. (You may find it useful to read the short ones too: [Beck *et al.* 2004a; 2004b].

References

Sigal Asaf, Haggai Eran, Yossi Richter, Daniel Connors, Donna Gresh, Julio Ortega, and Michael Mcinnis. Applying Constraint Programming to Identification and Assignment of Service Professionals. In David Cohen, editor, *Principles and Practice of Constraint Programming (CP 2010)*, volume 6308 of *Lecture Notes in Computer Science*, pages 24–37. Springer, 2010.

Albert Atserias, Andrei A. Bulatov, and Víctor Dalmau. On the Power of k-Consistency. In *34th International Colloquium on Automata, Languages and Programming (ICALP)*, volume 4596 of *LNCS*, pages 279–290. Springer, 2007.

Fahiem Bacchus and Kostas Stergiou. Solution Directed Backjumping for QCSP. In 13^{th} *International Conference on Principle and Practice of Constraint Programming (CP 07)*, volume 4741 of *Lecture Notes in Computer Science*, pages 148–163. Springer Verlag, 2007.

Andreas Bauer, Viorica Botea, Mark Brown, Matt Gray, Daniel Harabor, and John Slaney. An Integrated Modelling, Debugging, and Visualisation Environment for G12. In David Cohen, editor, *Principles and Practice of Constraint Programming (CP 2010)*, volume 6308 of *Lecture Notes in Computer Science*, pages 522–536. Springer, 2010.

J. Christopher Beck, Patrick Prosser, and Richard J. Wallace. Toward Understanding Variable Ordering Heuristics for Constraint Satisfaction Problems. In *Proceedings of the Fourteenth Irish Artificial Intelligence and Cognitive Science Conference (AICS03)*, 2003.

J. Christopher Beck, Patrick Prosser, and Richard J. Wallace. Failing First: An Update. In *Proceedings of the Sixteeth European Conference on Artificial Intelligence*, pages 959–960, 2004.

J. Christopher Beck, Patrick Prosser, and Richard J. Wallace. Variable Ordering Heuristics Show Promise. In *the Tenth International Conference on Principles and Practice of Constraint Programming (CP'04)*, 2004.

Belaid Benhamou. Study of Symmetry in Constraint Satisfaction Problems. In *Second Workshop on Principles and Practice of Constraint Programming*, pages 246–254, Orcas Island, Wa, 1994.

Christian Bessière, Jean-Charles Régin, Roland H.C. Yap, and Yuanlin Zhang. An Optimal Coarse-Grained Arc Consistency Algorithm. *Artificial Intelligence*, 165(2):165–185, 2005.

Christian Bliek and Djamilla Sam-Haroud. Path Consistency for Triangulated Constraint Graphs. In *Proc. of the 16 th IJCAI*, pages 456–461, Stockholm, Sweden, 1999.

Karl Blomdahl, Pierre Flener, and Justin Pearson. Contingency Plans for Air Traffic Management. In David Cohen, editor, *Principles and Practice of Constraint Programming (CP 2010)*, volume 6308 of *Lecture Notes in Computer Science*, pages 643–657. Springer, 2010.

Hadrien Cambazard. Basic Understanding of The Propagation Performed in a Constraint Solver Through the Kakuro Puzzle: Insights About Bound Consistency And Generalized Arc Consistency. http://4c.ucc.ie/ hcambaza/page1/page7/page7.html.

Assef Chmeiss and Philippe Jégou. Efficient Path-Consistency Propagation. *International Journal on Artificial Intelligence Tools*, 7(2):121–142, 1998.

Assef Chmeiss, Philippe Jégou, and Lamia Keddar. On a Generalization of Triangulated Graphs for Domains Decomposition of CSPs. In *Proc. of the 18 th IJCAI*, pages 203–208, Acapulco, Mexico, 2003.

Berthe Y. Choueiry and Amy M. Davis. Dynamic Bundling: Less Effort for More Solutions. In Sven Koenig and Robert Holte, editors, *5th International Symposium on Abstraction, Reformulation and Approximation (SARA 2002)*, volume 2371 of *Lecture Notes in Artificial Intelligence*, pages 64–82. Springer Verlag, 2002.

Hélène Collavizza and Michel Rueher. Exploring Different Constraint-Based Modelings for Program Verification. In 13^{th} *International Conference on Principle and Practice of Constraint Programming (CP 07)*, volume 4741 of *Lecture Notes in Computer Science*, pages 49–63. Springer Verlag, 2007.

Nadia Creignou, Phokion G. Kolaitis, and Heribert Vollmer, editors. *Complexity of Constraints*, volume 5250 of *Dagstuhl Seminar Proceedings, LNCS*, 2008.

Rina Dechter, Kalev Kask, Eyal Bin, and Roy Emek. Generating Random Solutions for Constraint Satisfaction Problems. In *Proc. of AAAI-2002*, pages 15–21, Edmoton, Alberta, Canada, 2002.

Rina Dechter. Constraint Processing. Manuscript, forthcoming, 2003.

Rina Dechter. *Constraint Processing*. Morgan Kaufmann, 2003.

Eugene C. Freuder and Charles D. Elfe. Neighborhood Inverse Consistency Preprocessing. In *Proc. of AAAI-96*, pages 202–208, Portland, Oregon, 1996.

Alfonso Gerevini and Alessandro Saetti. Efficient Computation of Minimal Point Algebra Constraints by Metagraph Closure. In 13^{th} *International Conference on Principle and Practice of Constraint Programming (CP 07)*, volume 4741 of *Lecture Notes in Computer Science*, pages 301–316. Springer Verlag, 2007.

Martin Grohe and Dániel Marx. Constraint Solving via Fractional Edge Covers. In *SODA '06: Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete algorithm*, pages 289–298. ACM, 2006.

Praveen Guddeti and Berthe Y. Choueiry. Characterization of a New Restart Strategy for Randomized Backtrack Search. In Boi Faltings et al., editor, *Recent Advances in Constraints*, volume 3419 of *Lecture Notes in Artificial Intelligence*, pages 56–70. Springer, 2005.

Samid Hoda, Willem Jan van Hoeve, and John N. Hooker. A systematic approach to mdd-based constraint programming. In *Principles and Practice of Constraint Programming - CP 2010 - 16th International Conference, CP 2010, St. Andrews, Scotland, UK, September 6-10, 2010. Proceedings*, volume 6308 of *Lecture Notes in Computer Science*, pages 266–280. Springer, 2010.

Víctor Dalmau Hubie Chen. Beyond Hypertree Width: Decomposition Methods Without Decompositions. In *11th International Conference on Principles and Practice of Constraint Programming (CP 05)*, volume 3709 of *LNCS*, pages 167–181. Springer, 2005.

Phokion G. Kolaitis and Moshe Y. Vardi. In Nadia Creignou, Phokion G. Kolaitis, and Heribert Vollmer, editors, *Complexity of Constraints*, volume 5250 of *Dagstuhl Seminar Proceedings, LNCS*, chapter A Logical Approach to Constraint Satisfaction, pages 125–155. Springer, 2008.

T. K. Satish Kumar. On the Tractability of Restricted Disjunctive Temporal Problems. In *ICAPS 2005*, pages 110–119. AAAI Press, 2005.

Y.C. Law, J.H.M. Lee, Toby Walsh, and J.Y.K. Yip. Breaking Symmetry of Interchangeable Variables and Values. In 13^{th} *International Conference on Principle and Practice of Constraint Programming (CP 07)*, volume 4741 of *Lecture Notes in Computer Science*, pages 423–437. Springer Verlag, 2007.

Christophe Lecoutre, Stéphane Cardon, and Julien Vion. Path Consistency by Dual Consistency. In 13^{th} *International Conference on Principle and Practice of Constraint Programming (CP 07)*, volume 4741 of *Lecture Notes in Computer Science*, pages 448–452. Springer Verlag, 2007.

Christophe Lecoutre, Lakhdar Sais, Sébastien Tabary, and Vincent Vidal. Exploiting Past and Future: Pruning by Inconsistent Partial State Dominance. In 13^{th} *International Conference on Principle and Practice of Constraint Programming (CP 07)*, volume 4741 of *Lecture Notes in Computer Science*, pages 453–467. Springer Verlag, 2007.

David Lesaint, Deepak Mehta, Barry O'Sullivan, Luis Quesada, and Nic Wilson. Context-Sensitive Call Control Using Constraints and Rules. In David Cohen, editor, *Principles and Practice of Constraint Programming (CP 2010)*, volume 6308 of *Lecture Notes in Computer Science*, pages 583–597. Springer, 2010.

Alan K. Mackworth. Consistency in Networks of Relations. *Artificial Intelligence*, 8:99–118, 1977.

Arnold Maestre and Christian Bessière. Improving Asynchronous Backtracking for Dealing with Complex Local Problems. In *European Conference on Artificial Intellifence (ECAI 04)*, pages 206–210, Valencia, Spain, 2004.

Pedro Meseguer and Carme Torras. Exploiting Symmetries Within Constraint Satisfaction Search. *Artif. Intell.*, 129((1-2)):133–163, 2001.

Laurent Michel and Pascal Van Hentenryck. Activity-Based Search for Black-Box Constraint Programming Solvers. In *CPAIOR*, volume 7298 of *Lecture Notes in Computer Science*, pages 228–243. Springer, 2012.

Philippe Refalo. Impact-Based Search Strategies for Constraint Programming. In *9th International Conference on Principles and Practice of Constraint Programming (CP 04)*, volume 3258 of *LNCS*, pages 557–571. Springer, 2004.

Jean-Charles Régin. Generalized Arc Consistency for Global Cardinality Constraint. In *Proceedings of the National Conference on Artificial Intelligence (AAAI 1996)*, pages 209–215, 1996.

Martin Sachenbacher and Brian C. Williams. Diagnosis as Semiring-Based Constraint Optimization. In *European Conference on Artificial Intellifence (ECAI 04)*, pages 873–877, Valencia, Spain, 2004.

András Z. Salamon and Peter G. Jeavons. Perfect Constraints Are Tractable. In 14^{th} *International Conference on Principle and Practice of Constraint Programming (CP 08)*, volume 5202 of *Lecture Notes in Computer Science*, pages 524–528. Springer, 2008.

Pierre Schaus, Yves Deville, and Pierre Dupont. Bound-Consistent Deviation Constraint. In 13^{th} *International Conference on Principle and Practice of Constraint Programming (CP 07)*, volume 4741 of *Lecture Notes in Computer Science*, pages 620–634. Springer Verlag, 2007.

Helmut Simonis, Paul Davern, Jacob Feldman, Deepak Mehta, Luis Quesada, and Mats Carlsson. A Generic Visualization Platform for CP. In David Cohen, editor, *Principles and Practice of Constraint Programming (CP 2010)*, volume 6308 of *Lecture Notes in Computer Science*, pages 460–474. Springer, 2010.

Helmut Simonis. Kakuro as a Constraint Problem. In *the* 7^7*th International Workshop on Constraint Modelling and Reformulation*, pages 1–15, 2008.

Barbara M. Smith and Paula Sturdy. Value Ordering for Finding All Solutions. In *International Joint Conference on Artificial Intelligence (IJCAI 2005)*, pages 311–316, 1997.

Michaël Soulignac, Michel Rueher, and Patrick Taillibert. A Safe and Flexible CP-Based Approach for Velocity Tuning Problems. In David Cohen, editor, *Principles and Practice of Constraint Programming (CP 2010)*, volume 6308 of *Lecture Notes in Computer Science*, pages 628–642. Springer, 2010.

Ioannis Tsamardinos and Martha E. Pollack. Efficient Solution Techniques for Disjunctive Temporal Reasoning Problems. *Artificial Intelligence*, 2003. In press.

Roman van der Krogt, Jacob Feldman, James Little, and David Stynes. An Integrated Business Rules and Constraints Approach to Data Centre Capacity Management. In David Cohen, editor, *Principles and Practice of Constraint Programming (CP 2010)*, volume 6308 of *Lecture Notes in Computer Science*, pages 568–582. Springer, 2010.

Toby Walsh. Depth-bounded Discrepancy Search. In *Proc. of the 15* th *IJCAI*, pages 1388–1395, Nagoya, Japan, 1997.

Huayue Wu and Peter van Beek. On Universal Restart Strategies for Backtracking Search. In $13th$ *International Conference on Principles and Practice of Constraint Programming (CP 05)*, volume 4741 of *Lecture Notes in Computer Science*, pages 681—695, 2007.

Stéphane Zampelli, Yves Deville, Christine Solmon, Sébastien Sorlin, and Pierre Dupont. Filtering for Subgraph Isomorphims. In 13^{th} *International Conference on Principle and Practice of Constraint Programming (CP 07)*, volume 4741 of *Lecture Notes in Computer Science*, pages 728–742. Springer Verlag, 2007.

Alessandro Zanarini and Gilles Pesant. Solution Counting Algorithms for Constraint-Centered Search Heuristics. In $13th$ *International Conference on Principles and Practice of Constraint Programming (CP 07)*, volume 4741 of *Lecture Notes in Computer Science*, pages 743–757. Springer, 2007.

Roie Zivan and Amnon Meisels. Dynamic Ordering for Asynchronous Backtracking on DisCSPs. In $11th$ *International Conference on Principles and Practice of Constraint Programming (CP 05)*, volume 3709 of *LNCS*, pages 32–46, 2005.

Roie Zivan, Moshe Zazone, and Amnon Meisels. Min-Domain Ordering for Asynchronous Backtracking. In $13th$ *International Conference on Principles and Practice of Constraint Programming (CP 07)*, volume 4741 of *Lecture Notes in Computer Science*, pages 758–772. Springer Verlag, 2007.