Computational Models of Change Propagation

Alexander V. Konstantinou
Computer Science Department
Columbia University in the City of New York

akonstan@cs.columbia.edu

Candidacy Exam
December 9th, 1999

1 Purpose

"The candidacy exam certifies that the student has demonstrated a depth of scholarship in the literature and the methods of the student's chosen area of research, and has demonstrated a facility with the scholarly skills of critical evaluation and verbal expression." (source : CS Dept. Candidacy Exam Requirements).

2 Candidate Research Area Statement

Network element and service configuration is currently a manual process. System administrators configure devices and services by editing files, using text based interactive shells, or through some forms based graphical user interface. Each of these methods entails different risks. For example, a manually edited file may contain syntax errors, and changes through a shell or a graphical interface may not be easily reversible. State of the art installations apply a combination of version control software and device/service-specific syntactical and semantic checkers (e.g. DNS lint) to maintain correctness through change.

The current methodology is not without its risks since users may subvert the version control system, or push through changes that were flagged as erroneous by the checker (the entire `.com` domain was inaccessible for over 24 hours when a Network Solutions employee ignored warnings and pushed through a corrupted DNS root record). Moreover, the syntax and semantic checkers must themselves be configured to enforce the local domain policies. Since each checker is custom developed for each device or service, administrators must learn several different languages for expressing configuration and policy constraints.

More importantly, this whole approach fails when configuration constraints must be enforced across devices or services. Currently, there is no alternative but to depend on the expertise of system administrators who must constantly maintain these high level constraints when making configuration changes. Sometimes, through human error, or an unforeseen new interaction between services, multi-device/service configuration changes may result in partial, or total network failure. Network administrators are then forced to undo these changes by individually restoring each configuration file. If the correct order is not maintained, some changes are not undone, or due to side effects which have made the previous network state unattainable or unstable, normal network service may still not be restored.

The goal our research is to provide core technologies for automating those aspects of network configuration. Our approach focuses on :

- Extending existing network modeling languages to support modeling of static and dynamic network device and service configuration. This unified modeling language will provide an abstraction from the underlying configuration mechanisms and simplify the task of system administrators. Additionally, the unified model will significantly simplify the task of defining cross device/service configuration constraints and change propagation,
- Creating a configuration policy language which will enable system administrators to express constraints on configuration states and configuration change propagation. System administrators will use this language to create simple scripts for performing complex configuration changes, such as, migrating a web server to a new subnet.
- Developing a configuration transaction system which will define the semantics of network device configuration transactions. Configuration scripts will execute as a transaction which may be aborted or rolled over.
- An architecture for integrating these novel technologies to existing networks along with a prototype implementation.

More background on the project, which is called NESTOR (NEtwork Self managemenT and ORganization) may be found in the URL :
`http://www.cs.columbia.edu/dcc/nestor/`

3 Exam Scope and Structure

The goal of the candidacy exam will be to assist the candidate in establishing depth of scholarship in the literature related to his research area. Although the main application of this research will be in network management, the non-architectural core research issues require a background in language design, constraint languages and systems, as well as transaction processing systems. Given the breadth of the material covered, and the candidate's previous experience in network management it is desirable to narrow the focus of the exam to those other areas.

For that purpose, the candidate has selected references that cover the area that can be broadly characterized as "computational models of change propagation". In accordance to the requirements outlined in the departmental candidacy exam requirements, the candidate will prepare a 30 minute oral presentation based on the selected references, followed by a 90 minute question and answer session.

4 Citations

- Theory: [Mac77] [MH86] [JL87] [Coh90] [MF90] [Mac92] [Zan92] [MJ92]
- Algorithms: [Kum92] [San95] [ZMGS94] [BAFB96] [Fru98]
- Interval Labels: [Dav87] [Hyv92]
- Constraint Hierarchies: [BFBW92] [HMY96]
- Systems/Imperative Constraint Programming: [APR90] [Wil91] [FBB92] [LFBB94] [Smo95] [SRF96]

References

[APR90] Paolo Avesani, Anna Perini, and Francesco Ricci. Cool: An object system with constraints. In Jean Bezivin, Bertrand Meyer, , and Jean-Marc Nerson, editors, *Technology of Object-Oriented Languages and Systems - Proceedings of the Second International Conference TOOLS 90*, pages 221–228, Paris, June 1990.

[BAFB96] Alan Borning, Richard Anderson, and Bjorn Freeman-Benson. Indigo: A local propagation algorithm for inequality constraints. In *Proceedings of the 1996 ACM Symposium on User Interface Software and Technology*, pages 128–136, 1996.

[BFBW92] Alan Borning, Bjorn Freeman-Benson, and Molly Wilson. Constraint hierarchies. *Lisp and Symbolic Computation*, 5(3):223–270, September 1992.

[Coh90] Jacques Cohen. Constraint logic programming languages. *Communications of the ACM*, 33(7):52–68, July 1990.

[Dav87] Ernest Davis. Constraint propagation with interval labels. *Artificial Intelligence*, 32(3):281–331, 1987.

[FBB92] Bjorn Freeman-Benson and Alan Borning. Integrating constraints with an object-oriented language. In *Proceedings of the 1992 European Conference on Object-Oriented Programming*, pages 268–286, June 1992.

[Fru98] Thom Fruühwirth. Theory and practice of constraint handling rules. *Journal of Logic Programming*, 37, October 1998.

[HMY96] Hiroshi Hosobe, Satoshi Matsuoka, and Akinori Yonezawa. Generalized local propagation: A framework for solving constraint hierarchies. In E. C. Freuder, editor, *Second International Conference on Principles and Practice of Constraint Programming (CP96)*, Lecture Notes in Computer Science, vol. 1118, pages 237–251. Springer-Verlag, August 1996.

[Hyv92] Eero Hyvönen. Constraint reasoning based on interval arithmetic: the tolerance prapagation approach. *Artificial Intelligence*, 58(1–3):71–112, 1992.

[JL87] Joxan Jaffar and Jean-Louise Lassez. Constraint logic programming. In *Proceedings of the 14th ACM Symposium on Principles of Programming Languages*, pages 111–119, Munich, Germany, January 1987. ACM Press.

[Kum92] Vipin Kumar. Algorithms for constraint-satisfaction problems: A survey. *Artificial Intelligence*, 13(1):32–44, 1992.

[LFBB94] Gus Lopez, Bjorn Freeman-Benson, and Alan Borning. Implementing constraint imperative programming languages: The kaleidosope'93 virtual machine. In *Proceedings of the 1994 ACM Conference on Object-Oriented Programming Systems, Languages*, pages 259–271. ACM, October 1994.

[Mac77] Alan K. Mackworth. Consistency in networks of relations. *Artificial Intelligence*, 8(1):99–118, 1977.

[Mac92] Alan K. Mackworth. The logic of constraint satisfaction. *Artificial Intelligence*, 58(1–3):3–20, 1992.

[MF90] Sanjay Mittal and Brian Falkenhainer. Dynamic constraint satisfaction problems. In *Proceedings of the 8th National Conference on Artificial Intelligence*, pages 25–32, Boston, MA, August 1990. AAAI Press.

[MH86] R. Mohr and T. C. Henderson. Arc and path consistency revisited. *Artificial Intelligence*, 28:225–233, 1986.

[MJ92] Steven Minton and Mark D. Johnston. Minimizing conflicts: a heuristic repair method for constraint satisfaction and scheduling problems. *Artificial Intelligence*, 58(1–3):161–205, 1992.

[San95] Michael Sannella. *Principles and practice of constraint programming : the Newport papers*, chapter The SkyBlue Constraint Solver and Its Applications. MIT Press, 1995.

[Smo95] Gert Smolka. The Oz programming model. In Jan van Leeuwen, editor, *Computer Science Today*, Lecture Notes in Computer Science, vol. 1000, pages 324–343. Springer-Verlag, Berlin, 1995.

[SRF96] Mihaela Sabin, Robert D. Russell, and Eugene C. Freuder. Constraint-based modeling: From diagnosis and configuration to network management. In *Proceedings of the CP96 Worshop on Applications of Constraint Programming*, Cambridge, MA, August 1996.

[Wil91] Michael Wilk. Equate: An object-oriented constraint solver. In *Proceedings of the 1991 ACM Conference on Object-Oriented Programming Systems, Languages, and Applications*, pages 286–298. ACM, October 1991.

[Zan92] Brad Vander Zanden. *Languages for Developing User Interfaces*, chapter An Active-Value-Spreadsheet Model for Interactive Languages, pages 183–210. Jones and Bartlett Publishers, Inc, 1992.

[ZMGS94] Brad T. Vander Zanden, Brad A. Myers, Dario A. Giuse, and Pedro A. Szekely. Integrating pointer variables into one-way constraint models. In *ACM Transactions on Computer-Human Interaction (TOCHI)*, volume 1, pages 161–213, June 1994.