University of Nebraska - Lincoln
DigitalCommons@University of Nebraska - Lincoln

CSE Conference and Workshop Papers | Computer Science and Engineering, Department of

1-1-2004

Evaluating Consistency Algorithms for Temporal Metric Constraints

Yang Shi
University of Nebraska-Lincoln, yshi@cse.unl.edu

Anagh Lal
University of Nebraska - Lincoln, alal@cse.unl.edu

Berthe Y. Choueiry
University of Nebraska - Lincoln, choueiry@cse.unl.edu

Follow this and additional works at: http://digitalcommons.unl.edu/cseconfwork

Part of the Computer Sciences Commons

Shi, Yang; Lal, Anagh; and Choueiry, Berthe Y., "Evaluating Consistency Algorithms for Temporal Metric Constraints" (2004). *CSE Conference and Workshop Papers*. Paper 165.
http://digitalcommons.unl.edu/cseconfwork/165

This Article is brought to you for free and open access by the Computer Science and Engineering, Department of at DigitalCommons@University of Nebraska - Lincoln. It has been accepted for inclusion in CSE Conference and Workshop Papers by an authorized administrator of DigitalCommons@University of Nebraska - Lincoln.

Shi, Lal & Choueiry in AAAI '04 Proceedings, AAAI 04 L.
Pages 970-971. Copyright 2004, AAAI. Used by permission.

Evaluating Consistency Algorithms for Temporal Metric Constraints

Yang Shi , Anagh Lal , and **Berthe Y. Choueiry**
Constraint Systems Laboratory
Computer Science & Engineering
University of Nebraska-Lincoln
Email: {yshi, alal, choueiry}@cse.unl.edu

Abstract

We study the performance of some known algorithms for solving the Simple Temporal Problem (STP) and the Temporal Constraint Satisfaction Problem (TCSP). In particular, we empirically compare the Bellman-Ford (BF) algorithm and its incremental version (incBF) by (Cesta & Oddi 1996) to the △STP of (Xu & Choueiry 2003a). Among the tested algorithms, we show that △STP is the most efficient for determining the consistency of an STP, and that incBF combined with the heuristics of (Xu & Choueiry 2003b) is the most efficient for solving the TCSP. We plan to improve △STP by exploiting incrementality as in incBF and other new incremental algorithms.

1 Introduction

Many planning and scheduling applications rely on an efficient handling of temporal information. We study two networks of metric constraints: the Simple Temporal Problem (STP) and the Temporal Constraint Satisfaction Problem (TCSP). An STP (Figure 1) is defined by a graph $G = (V, E, I)$ where V is a set of vertices t_i representing time points; E is a set of directed edges $e_{i,j}$ representing constraints between two time points t_i and t_j; and I is a set of constraint labels for the edges. A constraint label $I_{i,j}$ of edge $e_{i,j}$ is a *unique* interval $[a, b]$, a and $b \in \mathbb{R}$, and denotes a constraint of bounded difference $a \leq (t_j - t_i) \leq b$. We assume that there is at most one constraint between any two vertices e_i and e_j and that the constraint $e_{i,j}$ labeled $[a, b]$ can also be referred to as the constraint $e_{j,i}$ labeled $[-b, -a]$. A TCSP (Figure 2) is defined by a similar graph $G = (V, E, I)$, where each edge label $I_{i,j} = \{l^1_{ij}, l^2_{ij}, \ldots, l^k_{ij}\}$ is a *set* of disjoint intervals denoting a disjunction of constraints of bounded differences between t_i and t_j. The consistency of the STP can be determined in polynomial time. Solving the TCSP is **NP**-complete and can be done by expressing the TCSP as a meta-CSP and solving it with backtrack search (Dechter, Meiri, & Pearl 1991). Every node in the tree for solving the meta-CSP (Figure 3) is an STP that must be tested for consistency. Thus, it is important to solve the STP efficiently. In this paper, we compare empirically the performance of various algorithms for solving the STP and TCSP. We plan to extend our approach to the Disjunctive Temporal Problem (Stergiou & Koubarakis 2000).

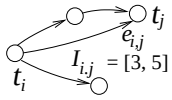

Figure 1: *STP*

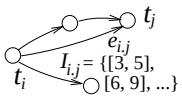

Figure 2: *TCSP*

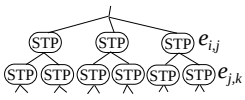

Figure 3: *Meta-CSP*

2 Background

The following algorithms can be used to determine the consistency of an STP: directional path consistency (DPC) (Dechter 2003), △STP (Xu & Choueiry 2003a), Floyd-Warshall (FW) and Bellman-Ford (BF) (Cormen, Leiserson, & Rivest 2001). In (Xu & Choueiry 2003a) we showed that △STP consistently outperforms FW, outperforms DPC on sparse graphs, and is comparable to DPC on dense graphs. We did not cover BF. Cesta and Oddi (1996) introduced an incremental version of BF (incBF) but did not test it on the TCSP. In (Xu & Choueiry 2003b) we showed that △STP outperforms FW and DPC on the TCSP, and proposed △STP-TCSP that dramatically improves search performance by combining △STP and two new heuristics (EdgeOrd and NewCyc). In this paper, we achieve two tasks: (1) Compare BF and △STP for solving the STP, and (2) Compare incBF and △STP for solving the TCSP.

3 Algorithms tested & experiments

A known technique for enhancing the performance of solving a Constraint Satisfaction Problem (CSP) is to decompose the graph of the CSP into biconnected components according to its articulation points[1], independently solve each component, then combine the results. This technique (AP) provides an upper bound, in the size of the largest biconnected component, to the search effort (Freuder 1985). We integrated AP in each one of the tested solvers except for △STP in which AP is implicit. For the TCSP, we used △AC of (Choueiry & Xu 2004) as a preprocessing step for filtering the constraints. We compared the algorithms listed in Table 1 for solving the STP and the TCSP on randomly generated problems, measuring averages of CPU time, the number of constraint checks `CC`, and the number of nodes visited `NV` (for TCSP). We used the random generators of

Copyright © 2004, American Association for Artificial Intelligence (www.aaai.org). All rights reserved.

[1] An articulation point of a graph is a vertex whose removal disconnects the graph into its biconnected components.

(Xu & Choueiry 2003b), varying constraint density d in [2%, 90%], where $d = \frac{|E|-e_{min}}{e_{max}-e_{min}}$, e_{max}=$(\frac{|V|(|V|-1)}{2})$, and e_{min}=$(|V|-1)$. Below we summarize the three experiments, which lead to the ranking shown in Table 1. The first experiment compares STP solvers. We used STP instances of 50 nodes and generated 100 instances per density value. The results are shown in Figure 4. The second experiment compares TCSP solvers. We used problems with 10 nodes and 3 to 5 intervals per node, and generated 600 instances per density value. (The size of the meta-CSP grows exponentially in the number of nodes in the TCSP, thus limiting the size of tested instances.) We tested finding both the first solution and all solutions to the TCSP (i.e., the minimal network), but report in Figure 5 only the latter because the difference in behavior is more significant. Similar results hold for NV and CPU time. Table 2 shows the average CC gain of the best strategy and its lower limit (LL) and upper limit (UL) with 95% confidence, where the gain is the difference of the values for the last two algorithms in Table 1. Table 2 shows a high instability for $d \in [2\%, 10\%]$. The third experiment justifies the unreliability of these results. We computed the cumulative averages of the performance measures as the sample size increases. Figure 6 shows that, for small values of d, the average of CC is not stable when the sample size is below 400 samples. Hence results for these density values are not statistically significant.

Algorithm	STP	TCSP
FW+AP	worse	worse
DPC+AP	good	OK
BF+AP	OK	–
△STP	best	–
incBF+AP	–	good
△STP+EdgeOrg+NewCyc	–	better
incBF+AP+EdgeOrg+NewCyc *(new)*	–	best

Table 1: *Ranking according to performance.*

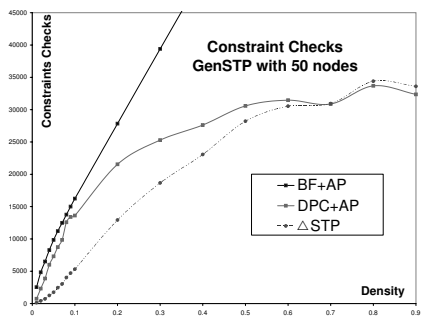

Figure 4: *Constraint checks for selected STP solvers.*

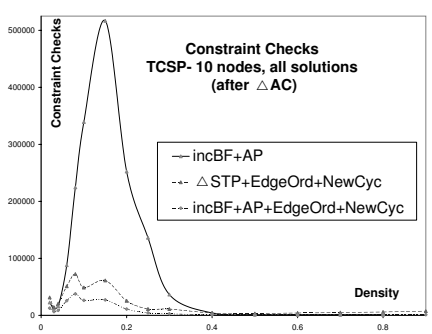

Figure 5: *Constraint checks for selected TCSP solvers.*

d	△ STP CC$\times 10^3$	IncBF CC$\times 10^3$	Gain CC$\times 10^3$ LL	Average	UL
0.02	45.61	14.77	5.39	30.84	56.29
0.04	17.51	7.56	5.06	9.95	14.84
0.06	51.66	24.30	3.45	27.35	51.24
0.08	83.38	50.74	4.86	32.63	60.41
0.10	50.31	26.24	20.29	24.07	27.84
0.15	75.92	37.61	20.52	38.30	56.08
0.20	28.09	12.03	10.74	16.06	21.38

Table 2: *incBF+AP vs. △STP, both including EdgeOrg+NewCyc.*

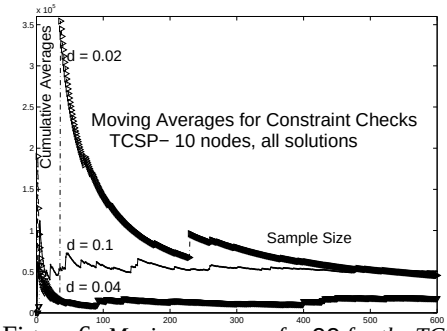

Figure 6: *Moving averages for* CC *for the TCSP.*

4 Conclusions

We tested empirically the behavior of various algorithms for solving temporal networks. We showed that the performance of incBF for solving the TCSP can be significantly improved when combined with the heuristics proposed in (Xu & Choueiry 2003b). In the future, we plan to exploit the incrementality feature of incBF and of other new algorithms indicated to us by the reviewers. This research is supported by NSF CAREER Award #0133568.

References

Cesta, A., and Oddi, A. 1996. Gaining Efficiency and Flexibility in the Simple Temporal Problem. In *TIME 96*.

Choueiry, B., and Xu, L. 2004. An Efficient Consistency Algorithm for the Temporal Constraint Satisfaction Problem. In *Submitted to AI Communications*.

Cormen, T.; Leiserson, C.; and Rivest, R. L. 2001. *Introduction to Algorithms*. McGraw-Hill & MIT Press.

Dechter, R.; Meiri, I.; and Pearl, J. 1991. Temporal Constraint Networks. *AIJ* 49:61–95.

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

Freuder, E. 1985. A Sufficient Condition for Backtrack-Bounded Search. *JACM* 32 (4):755–761.

Stergiou, K., and Koubarakis, M. 2000. Backtracking Algorithms for Disjunctions of Temporal Constraints. *AIJ* 12- (1):81–117.

Xu, L., and Choueiry, B. 2003a. A New Efficient Algorithm for Solving the Simple Temporal Problem. In *TIME-ICTL 03*, 212–222. IEEE Press.

Xu, L., and Choueiry, B. 2003b. Improving Backtrack Search for Solving the TCSP. In *CP 03*, 754–768. LNCS 2833.