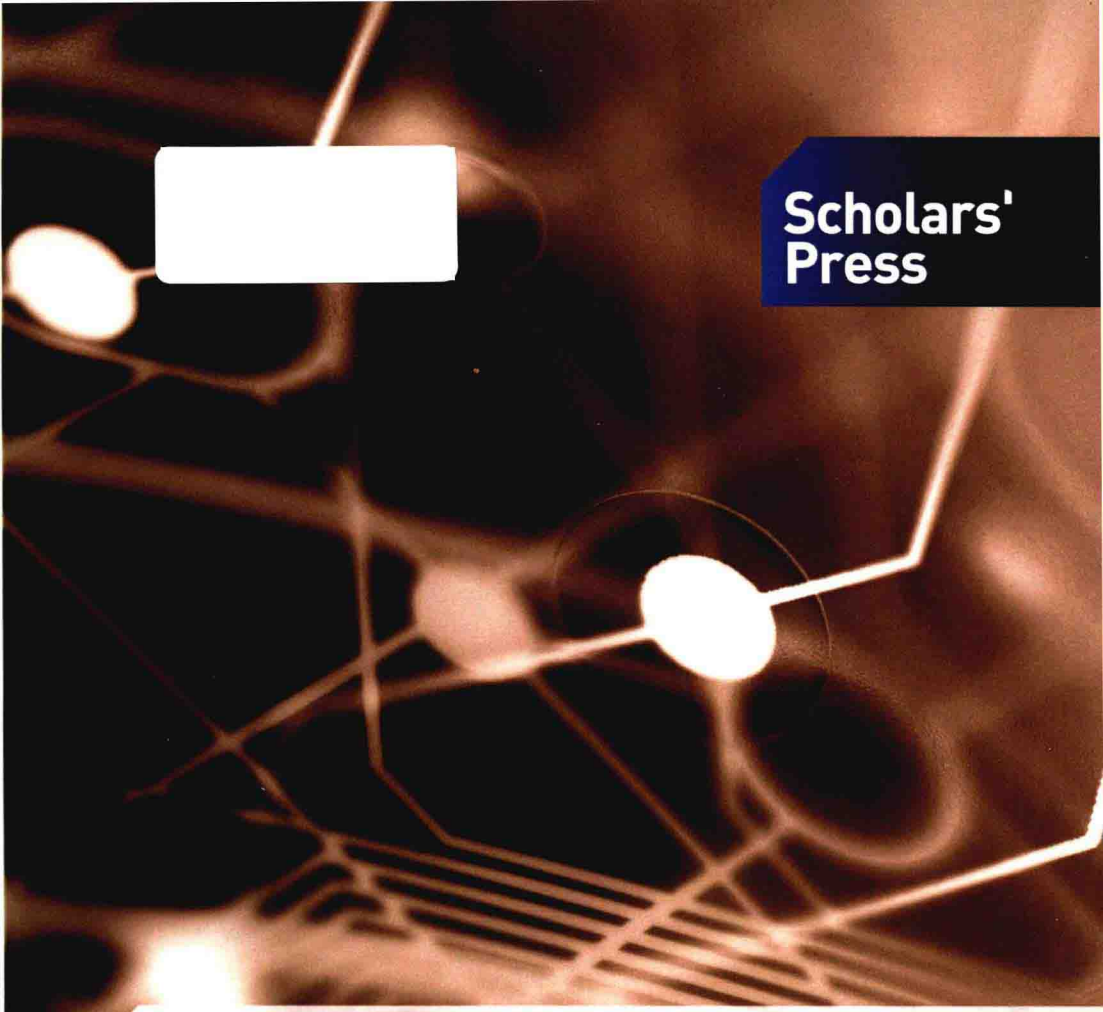


Scholars' Press

William Yeoh

Speeding Up Distributed Constraint Optimization Search Algorithms

William Yeoh

Speeding Up Distributed Constraint Optimization Search Algorithms

Scholar's Press

Impressum / Imprint
Bibliografische Information der Deutschen Nationalbibliothek: Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über http://dnb.d-nb.de abrufbar.

Bibliographic information published by the Deutsche Nationalbibliothek: The Deutsche Nationalbibliothek lists this publication in the Deutsche Nationalbibliografie; detailed bibliographic data are available in the Internet at http://dnb.d-nb.de.

Coverbild / Cover image: www.ingimage.com

Verlag / Publisher:
Scholar's Press
ist ein Imprint der / is a trademark of
OmniScriptum GmbH & Co. KG
Heinrich-Böcking-Str. 6-8, 66121 Saarbrücken, Deutschland / Germany
Email: info@scholars-press.com

Herstellung: siehe letzte Seite /
Printed at: see last page
ISBN: 978-3-639-70721-2

Zugl. / Approved by: Los Angeles, University of Southern California, Ph.D. Dissertation, 2010

William Yeoh

Speeding Up Distributed Constraint Optimization Search Algorithms

Table of Contents

Abstract

Distributed constraint optimization (DCOP) is a model where several agents coordinate with each other to take on values so as to minimize the sum of the resulting constraint costs, which are dependent on the values of the agents. This model is becoming popular for formulating and solving agent-coordination problems. As a result, researchers have developed a class of DCOP algorithms that use search techniques. For example, Asynchronous Distributed Constraint Optimization (ADOPT) is one of the pioneering DCOP search algorithms that has been widely extended. Since solving DCOP problems optimally is NP-hard, solving large problems efficiently becomes an issue.

DCOP search algorithms can be viewed as distributed versions of centralized search algorithms. Therefore, I hypothesize that one can speed up DCOP search algorithms by applying insights gained from centralized search algorithms, specifically (1) by using an appropriate search strategy, (2) by sacrificing solution optimality, (3) by using more memory, and (4) by reusing information gained from solving similar DCOP problems. However, DCOP search algorithms are sufficiently different from centralized search algorithms that these insights cannot be trivially applied.

To validate my hypotheses: (1) I introduce Branch-and-Bound ADOPT (BnB-ADOPT), an extension of ADOPT that changes the search strategy of ADOPT from memory-bounded best-first search to depth-first branch-and-bound search, resulting in one order of magnitude speedup. These results validate my hypothesis that DCOP search algorithms that employ depth-first branch-and-bound search can be faster than DCOP search algorithms that employ memory-bounded best-first search. (2) I introduce an approximation mechanism that uses weighted heuristic values to trade off solution costs for smaller runtimes. This approximation mechanism allows ADOPT and BnB-ADOPT to terminate faster with larger weights, validating my hypothesis that DCOP search algorithms that use weighted heuristic values can have runtimes

that decrease as larger weights are used. Additionally, the new approximation mechanism provides relative error bounds and thus complements existing approximation mechanisms that only provide absolute error bounds. (3) I introduce the MaxPriority, MaxEffort and MaxUtility DCOP-specific caching schemes, which allow ADOPT and BnB-ADOPT to cache DCOP-specific information when they have more memory available and terminate faster with larger amounts of memory. Experimental results show that the MaxEffort and MaxUtility schemes speed up ADOPT more than the currently used generic caching schemes, and the MaxPriority scheme speeds up BnB-ADOPT at least as much as the currently used generic caching schemes. Therefore, these results validate my hypothesis that DCOP-specific caching schemes can reduce the runtime of DCOP search algorithms at least as much as the currently used generic caching schemes. (4) I introduce an incremental procedure and an incremental pseudo-tree reconstruction algorithm that allow ADOPT and BnB-ADOPT to reuse information gained from solving similar DCOP problems to solve the current problem faster, resulting in runtimes that decrease with larger amounts of information reuse. These results validate my hypothesis that DCOP search algorithms that reuse information from searches of similar DCOP problems to guide their search can have runtimes that decrease as they reuse more information.

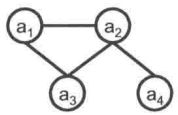
Chapter 1

Introduction

Distributed constraint optimization (DCOP) (Modi, 2003; Mailler, 2004; Petcu, 2007; Pearce, 2007; Burke, 2008) is a model where several agents coordinate with each other to take on values so as to minimize the sum of the resulting constraint costs, which are dependent on the values of the agents. This model is becoming popular for formulating and solving agent-coordination problems (Lesser, Ortiz, & Tambe, 2003; Maheswaran, Pearce, & Tambe, 2004; Schurr, Okamoto, Maheswaran, Scerri, & Tambe, 2005; Junges & Bazzan, 2008; Ottens & Faltings, 2008; Kumar, Faltings, & Petcu, 2009). As a result, researchers have developed several DCOP algorithms that use search techniques. For example, Asynchronous Distributed Constraint Optimization (ADOPT) (Modi, Shen, Tambe, & Yokoo, 2005) is one of the pioneering DCOP search algorithms that has been widely extended. Since solving DCOP problems optimally is NP-hard (Modi et al., 2005), solving large problems efficiently becomes an issue.

DCOP search algorithms can be viewed as distributed versions of centralized search algorithms. Therefore, I hypothesize that one can speed up DCOP search algorithms by applying insights gained from centralized search algorithms, specifically (1) by using an appropriate search strategy, (2) by sacrificing solution optimality, (3) by using more memory, and (4) by reusing information from searches of similar DCOP problems.

To validate my hypotheses: (1) I introduce Branch-and-Bound ADOPT (BnB-ADOPT), an extension of ADOPT that changes the search strategy of ADOPT from memory-bounded best-first search to depth-first branch-and-bound search, resulting in one order of magnitude speedup when solving sufficiently large DCOP problems; (2) I introduce an approximation mechanism that allows ADOPT and BnB-ADOPT



(a)

a_1	a_2	Cost
0	0	5
0	1	8
1	0	20
1	1	3

a_1	a_3	Cost
0	0	5
0	1	10
1	0	20
1	1	3

a_2	a_3	Cost
0	0	5
0	1	4
1	0	3
1	1	3

a_2	a_4	Cost
0	0	3
0	1	8
1	0	10
1	1	3

(b)

Figure 1.1: Example DCOP Problem

to use weighted heuristic values to trade off solution costs for smaller runtimes; (3) I introduce DCOP-specific caching schemes that allow ADOPT and BnB-ADOPT to store more information when they have more memory available, which can be at least as fast as the currently used generic caching schemes; and (4) I introduce an incremental procedure and an incremental pseudo-tree reconstruction algorithm that allow ADOPT and BnB-ADOPT to reuse information from searches of similar DCOP problems. ADOPT and BnB-ADOPT terminate faster when they reuse more information.

1.1 DCOP Problems

A DCOP problem consists of a set of agents, each responsible for taking on (= assigning itself) a value from its finite domain. The agents coordinate their value assignments, which are subjected to a set of constraints. Two agents are said to be constrained if they share a constraint. Each constraint has an associated cost, which depends on the values taken on by the constrained agents. An agent only knows the costs of constraints that it is involved in. A complete solution is an assignment of values to all agents, and a partial solution is an assignment of values to a subset of agents. The cost of a solution is the sum of the constraint costs of all constraints resulting from the given value assignments. Solving a DCOP problem optimally means to find a complete solution such that the sum of all constraint costs is minimized. Finding such a cost-minimal solution is NP-hard (Modi et al., 2005). It is common to visualize a DCOP problem as a constraint graph where the vertices are the agents and the edges are the constraints. Figure 1.1(a) shows the constraint graph of an example DCOP problem with four agents that can each take on the value zero

or one, and Figure 1.1(b) shows the constraint costs. The cost-minimal solution of our example DCOP problem is where all agents take on the value one, incurring a total cost of 12 (3 from each constraint).

The DCOP model is becoming popular for formulating and solving agent-coordination problems such as the distributed scheduling of meetings (Maheswaran et al., 2004; Petcu & Faltings, 2005b; Greenstadt, Grosz, & Smith, 2007; Zivan, 2008; Yeoh, Felner, & Koenig, 2009, 2010), the distributed coordination of unmanned aerial vehicles (Schurr et al., 2005), the distributed coordination of sensors in a network (Lesser et al., 2003; Zhang, Xing, Wang, & Wittenburg, 2003; Yeoh, Sun, & Koenig, 2009a; Yeoh, Varakantham, & Koenig, 2009b; Zivan, Glinton, & Sycara, 2009; Lisý, Zivan, Sycara, & Péchoucek, 2010), the distributed allocation of resources in disaster evacuation scenarios (Carpenter, Dugan, Kopena, Lass, Naik, Nguyen, Sultanik, Modi, & Regli, 2007; Lass, Kopena, Sultanik, Nguyen, Dugan, Modi, & Regli, 2008), the distributed synchronization of traffic lights (Junges & Bazzan, 2008), the distributed planning of truck routes (Ottens & Faltings, 2008), the distributed management of power distribution networks (Kumar et al., 2009) and the distributed generation of coalition structures (Ueda, Iwasaki, & Yokoo, 2010). As a result, researchers have developed several DCOP algorithms that use search techniques (= DCOP search algorithms). For example, ADOPT (Modi et al., 2005) is one of the pioneering DCOP search algorithms that has been widely extended (Modi & Ali, 2004; Ali, Koenig, & Tambe, 2005; Bowring, Tambe, & Yokoo, 2006; Davin & Modi, 2006; Pecora, Modi, & Scerri, 2006; Choxi & Modi, 2007; Matsui, Silaghi, Hirayama, Yokoo, & Matsuo, 2008; Silaghi & Yokoo, 2009; Matsui, Silaghi, Hirayama, Yokoo, & Matsuo, 2009; Gutierrez & Meseguer, 2010). ADOPT is a distributed best-first search algorithm that is complete and memory-bounded.

1.2 Hypotheses

DCOP search algorithms can be viewed as distributed versions of centralized search algorithms. Therefore, my hypothesis is as follows:

> *One can speed up DCOP search algorithms by applying insights gained from centralized search algorithms to DCOP search algorithms.*

Specifically, there are four common approaches that are used to speed up centralized search algorithms in the literature that can be applied to DCOP search algorithms:

1. One can speed up DCOP search algorithms by using an appropriate search strategy for the given problem type. A common approach is to use depth-first branch-and-bound search instead of memory-bounded best-first search for problems whose search trees are bounded. Researchers have shown that depth-first branch-and-bound search is faster than memory-bounded best-first search for these problems (Zhang & Korf, 1995). Therefore, I hypothesize that DCOP search algorithms that employ depth-first branch-and-bound search can be faster than DCOP search algorithms that employ memory-bounded best-first search since the search trees of DCOP problems are bounded.

2. One can speed up DCOP search algorithms by sacrificing solution optimality. A common approach is to use weighted heuristic values to focus the search. Algorithms that use this approach include Weighted A* (Pohl, 1970) and Weighted A* with dynamic weights (Pohl, 1973). These algorithms guarantee that the costs of the solutions found are at most a constant factor larger than the minimal costs, where the constant is the largest weight used. Typically, the runtime of these algorithms decreases as larger weights are used. Therefore, I hypothesize that DCOP search algorithms that use weighted heuristic values can have runtimes that decrease as larger weights are used.

3. One can speed up DCOP search algorithms by using more memory. A common approach is to cache information as long as memory is available, such that the cached information can be used when needed. Algorithms that use this approach include MA* (Chakrabarti, Ghosh, Acharya, & DeSarkar, 1989) and SMA* (Russell, 1992). Typically, the runtime of these algorithms decreases as more memory is available. Motivated by these results, researchers have developed any-space versions of DCOP search algorithms, such as any-space ADOPT (Matsui, Matsuo, & Iwata, 2005) and any-space NCBB (Chechetka & Sycara, 2006a), and showed that the runtime of these algorithms indeed decreases as more memory is available. However, they use generic caching

schemes, such as FIFO and LRU, that are similar to popular page replacement schemes used in operating systems. These generic schemes do not exploit the cached information in a DCOP-specific way. Therefore, I hypothesize that DCOP-specific caching schemes can reduce the runtime of DCOP search algorithms at least as much as the currently used generic caching schemes.

4. One can speed up DCOP search algorithms by reusing information from searches of similar DCOP problems. A common approach is to reuse information from searches of similar problems to guide the search. Algorithms that use this approach include incremental search algorithms (Koenig, Likhachev, Liu, & Furcy, 2004b) such as D* (Stentz, 1995), Adaptive A* (Koenig & Likhachev, 2005) and FRA* (Sun, Yeoh, & Koenig, 2009b). Typically, the runtime of these algorithms decreases as they reuse more information. Therefore, I hypothesize that DCOP search algorithms that reuse information from searches of similar DCOP problems to guide their search can have runtimes that decrease as they reuse more information.

1.3 Contributions

Although DCOP search algorithms can be viewed as distributed versions of centralized search algorithms, they are often independently developed. For example, ADOPT was developed independent of RBFS (Korf, 1993), but both algorithms share many properties. For example, both algorithms are memory-bounded, use the same search strategy and use the same principle to restore information already purged from memory. Therefore, one should be able to utilize the insights gained from centralized search algorithms to speed up DCOP search algorithms. However, DCOP search algorithms are sufficiently different from centralized search algorithms that these insights cannot be trivially applied. For example, unlike centralized search algorithms, DCOP search algorithms operate in a distributed fashion, have memory that is distributed among the agents and have agents that can only perform local searches, yet must follow a global search strategy.

This dissertation uses the four approaches described in Section 1.2 to speed up DCOP search algorithms. I make a design choice to use the framework of ADOPT,

which is one of the pioneering DCOP search algorithms, as the starting platform for the work in this dissertation. The motivation for this decision is that ADOPT has been widely extended (Modi & Ali, 2004; Ali et al., 2005; Bowring et al., 2006; Davin & Modi, 2006; Pecora et al., 2006; Choxi & Modi, 2007; Matsui et al., 2008; Silaghi & Yokoo, 2009; Matsui et al., 2009; Gutierrez & Meseguer, 2010) in addition to it having good properties. For example, agents in ADOPT operate concurrently and asynchronously to solve subproblems in parallel, which results in smaller runtimes than if they are to operate sequentially and synchronously. This dissertation makes the following four contributions:

1. To assess the hypothesis that DCOP search algorithms that employ depth-first branch-and-bound search can be faster than DCOP search algorithms that employ memory-bounded best-first search, we introduce BnB-ADOPT. BnB-ADOPT is a DCOP search algorithm that uses the framework of ADOPT but changes the search strategy of ADOPT from memory-bounded best-first search to depth-first branch-and-bound search. Although there exist other DCOP search algorithms, such as SBB (Hirayama & Yokoo, 1997), NCBB (Chechetka & Sycara, 2006b) and AFB (Gershman, Meisels, & Zivan, 2009), that employ depth-first branch-and-bound search, it is difficult to determine if depth-first branch-and-bound search is faster than memory-bounded best-first search since these algorithms differ by more than their search strategies when compared to ADOPT. In fact, SBB has been shown to be slower than ADOPT (Modi et al., 2005) while NCBB and AFB have been shown to be faster than ADOPT (Chechetka & Sycara, 2006b; Gershman et al., 2009). Hence, we introduce BnB-ADOPT since these results make clear the need for two DCOP search algorithms that differ *only* in their search strategies. Experimental results show that BnB-ADOPT is up to one order of magnitude faster than ADOPT when solving sufficiently large DCOP problems. Therefore, these results validate the hypothesis that DCOP search algorithms that employ depth-first branch-and-bound search can be faster than DCOP search algorithms that employ memory-bounded best-first search.

   This work is non-trivial since ADOPT is a rather complicated distributed algorithm whose agents operate concurrently and asynchronously at all times.