Interdependent Subproblems in Distributed Problem Solving

COINS Technical Report 93-58

Dorothy L. Mammen

Victor R. Lesser

Department of Computer Science

University of Massachusetts, Amherst, MA

Abstract

Using a distributed constraint satisfaction problem (DCSP) model of distributed problem solving, we define a new texture measure, *Imbalance in Variable Tightness*, or IVT, unique to distributed systems. IVT is a measure of unevenness in distribution of local measures of variable tightness, a CSP texture measure, among agents. A non-zero IVT can be *inherent*, caused by a skewed distribution of variables among agents with respect to their global variable tightness, or *information-based*, caused by the limited information of local views. We find that variable ordering based on variable tightness, a heuristic that works well in a centralized context, commonly breaks down in a distributed system involving parallel asynchronous agents, even when IVT = 0. Closer examination of *positive* and *negative* constraints among variables leads to specification of precedence relationships among particular pairs of variables, the collective use of which is usually as good as variable ordering in a distributed system based on global variable tightness. We find the significance of particular precedence relations to be highly context-dependent: a pairwise ordering that is highly beneficial early in problem solving can be highly detrimental later on. This has implications for task decomposition and coordination in distributed problem solving.

The project or effort depicted was or is sponsored by the Department of the Navy, Office of the Chief of Naval Research, Grants N00014-92-J-1698 and N00014-92-J-1450. The content of the information does not necessarily reflect the position or the policy of the Government, and no official endorsement should be inferred.

1 Introduction: The Problem

Finding solutions for large problems is sometimes most advantageously carried out as the joint responsibility of multiple agents. For example, a group of identical or similar agents might be employed purely for the purpose of finding the solution faster than can be accomplished by a single problem-solver. Alternatively, a multiple-agent approach can be essential when the areas of expertise relevant for solving the problem do not reside in any single agent. In either type of scenario, a problem is distributed in some way among multiple agents who, within some framework of interaction, find solutions to parts of the problem, and whose partial results are somehow combined into an overall solution.

It has been amply demonstrated that when subproblems of multiple agents are interdependent, the way in which they find, share, and use partial results can greatly affect the overall efficiency of the problem solving effort. One example of this phenomenon is *distraction*, in which an agent takes longer or fails to find a solution to its part of the problem due to relying on an incorrect partial result shared by another agent. Examples of distraction are plentiful. In decentralized job-shop scheduling, agents with order sets that are difficult to schedule can be forced into an infeasible scheduling situation by reservations made earlier by agents with easy order sets, even when a feasible scheduling solution exists [Sycara, Roth, Sadeh, and Fox 1991]. In the parametric design of a steam condenser, agents that are less constrained by the initial problem specification can slow down problem solving by distracting other agents with non-useful proposals [Lander and Lesser 1992]. In a distributed version of the Hearsay-II speech understanding system, agents with weak constraints could quickly generate and transmit incorrect partial results, which a receiving agent would try to extend [Lesser and Erman 1980]. A similar phenomenon was also seen in the Distributed Vehicle Monitoring Testbed, DVMT [Corkill and Lesser 1983].

In this work we examine how subproblem interactions can affect distributed problem solving efficiency. We take a distributed constraint satisfaction problem (DCSP) view of cooperative distributed problem solving (CDPS), for the same reasons as given by [Yokoo, Durfee, Ishida, and Kuwabara 1992]: because many CDPS problems can effectively be cast as DCSP, in which a set of variables are to receive assignments consistent with a set of constraints among them, and because DCSP provides a formal framework within which to study CDPS. This work is part of a larger project relating problem structure and problem solving strategies and efficiency. Such a theory can be used to direct the choice of strategy for solving a particular problem, along dimensions relating to cooperative control: what subproblems an agent should work on and when, to whom partial subproblem solutions or local constraints should be communicated and when, what credence should be given to external partial subproblem results or constraints, and so on [Lesser 1991]. In addition, the theory can be used to suggest what information needs to be collected about a problem in order to make a wise choice. Finally, the theory can also be used as a basis for heuristics to be employed when it is impossible or computationally infeasible to gather all the information necessary to make a completely informed choice.

In Section 2.1 we introduce the problem of variable ordering based on variable tightness in a distributed system, and in Section 2.2 we define and discuss a texture measure for distributed systems, called *Imbalance in Variable Tightness* (IVT), that describes the distribution of variables among agents with respect to global variable tightness. We work through an example in Section 2.3 and evaluate IVT as a predictor of distributed problem solving efficiency in Section 2.4. Finding that very small deviations from global variable tightness ordering can have a dramatic effect on problem solving efficiency, we move on to closer analysis of relationships among variables in Section 3. We predict and evaluate the significance of precedence relationships based on *negative* and *positive constraints*, and show that significance of some precedence relationships depends on problem solving context. In Section 4 we discuss implications of these findings and directions we are taking this work. Finally, the Appendix lists the scheduling problems that we used for these analyses.

2 A First Approach Using Textures

2.1 Variable and Value Ordering in Centralized and Distributed Systems

Brute force algorithms for solving constraint satisfaction problems, or CSP's, are exponential in worst case complexity, potentially involving a great deal of backtracking to determine a consistent set of variable assignments. Various heuristics have been suggested for reducing, on average, the amount of backtracking required. These heuristics [Dechter and Pearl 1988; Minton, Johnston, Philips, and Laird 1990; Purdom 1983] have focused on variable ordering, that is, heuristically ordering the instantiation of variables in order to reduce the amount of search necessary, and value ordering, that is, once a variable has been chosen, heuristically choosing a value that is more likely to be consistent with future assignments than to lead to an inconsistency that will necessitate backtracking.

Constraint satisfaction can be viewed as a search in which problem space states are constraint graphs, operators are variable assignments, and texture measures of the search space are used to heuristically focus search [Fox, Sadeh, and Baykan 1989]. The *variable tightness* texture, defined as the probability for each variable of having to backtrack to find a consistent solution if that variable is instantiated last, is one measure that can be used for variable ordering. Similarly, the *value goodness* texture, defined as the probability that assignment of a particular value leads to an overall solution to the problem, can be used for value ordering. Estimates of these measures have been shown to contribute to problem solving efficiency in the domains of spatial planning and factory scheduling [Fox et al. 1989].

In distributed versus centralized CSP, each agent has only part of the overall problem to solve. In addition, each is aware of only some of the constraints underlying the entire problem, namely those that are local to the agent and those that are part of the overall problem specification, while constraints local to other agents are generally not known. If the system is one in which agents asynchronously and in parallel carry out variable and value ordering based on local knowledge of constraints, and if subproblems are at all interdependent, then the distribution of tasks and constraint knowledge among agents can greatly affect problem solving efficiency. In particular, variables can be assigned either earlier or later than they would be based on a centralized view of constraints, and values may be chosen unwisely for lack of information. The resulting discrepancy from variable and value ordering based on a centralized view increases the probability that inter-agent backtracking will be necessary to find a consistent solution, or, if inter-agent backtracking is prohibited or limited, that a globally consistent solution will not be found.

2.2 Imbalance in Variable Tightness: A DCSP Texture Measure

We would like to characterize the conditions under which the order of assignments made in a system of distributed, asynchronous agents using variable ordering based on local views will match the order that would obtain in a centralized system. To minimize the discrepancy in these orderings, two conditions must hold:

(1) the variable tightness texture measures of the variables must have the same rank ordering in the distributed case, based on constraints known to the assigning agent, as over the entire set of constraints, and

(2) the variables to be assigned must be distributed evenly among the agents with respect to the global (centralized view) variable tightness rankings.

If we assume that agents acting asynchronously and in parallel assign variables at approximately the same rate and that the conditions above are met, then the resulting variable ordering will approximate as closely as possible for an asynchronous system the ordering that would be obtained by considering all constraints together. An exact match cannot be guaranteed with asynchronous agents, however, as we will see.

Directly verifying either of the above conditions in a distributed system requires knowing the variable tightness ranking of the variables under the full set of constraints. If both conditions (1) and (2) are met, however, then the distributions of locally-computed variable tightness measures should not differ significantly among the agents, whereas if either of the conditions is not met, then differences are likely. We define a new texture measure, called ***imbalance in variable tightness***, or ***IVT***, to be the extent to which variable tightness measures differ in distribution (mean and variance) among agents. Homogeneity of variances can be tested using an F-test for two agents or Bartlett's test for homogeneity of variances for more than two agents [Sokal and Rohlf 1981]. If the variances are not significantly different, then differences among means should be tested; this can be done using a t-test (two agents) or analysis of variance (more than two agents). Since the t-test and analysis of variance depend on homogeneous variances, they should not be carried out if the variances were found to be significantly different; this is fine, however, because in that case a high IVT has been detected already, anyway. IVT is defined exactly as follows:

For two agents:

$$\mathrm{IVT} = \begin{cases} 1 - p(F_s) & p(F_s) < 0.05 \\ 1 - \min\left[p(F_s),\ p(t_s)\right] & \text{otherwise} \end{cases}$$

For more than two agents:

$$\mathrm{IVT} = \begin{cases} 1 - p\left(X^2\right) & p\left(X^2\right) < 0.05 \\ 1 - \min\left[p\left(X^2\right),\ p\left(F_s\right)\right] & \text{otherwise} \end{cases}$$

where F_s is the statistic generated in an F-test and in analysis of variance, t_s is Student's t-statistic, and X^2 is the statistic generated in Bartlett's test for homogeneity of variances. IVT is an *extended* texture measure in that it is uniquely appropriate for distributed problem solving, because it specifically takes into account the existence of multiple local search spaces that can differ in their local texture measures.

A non-zero IVT can result from two different situations, which we distinguish as *inherent* and *information-based*. Firstly, it may be that variables are not distributed evenly among agents with respect to global variable tightness measures; that is, some agents have an "easier" set of variables to assign than others. This is a violation of Condition 2 above. In this case, the interleaving of assignments among asynchronous agents with easier and harder task sets results in some looser variables being assigned before tighter ones. We call this situation an *inherent* imbalance since it is caused by the actual distribution of tasks. It requires relatively strong measures to remedy, such as redistribution of the variables among the agents, use of a priority system for assignments and backtracking reassignments, or synchronization of agents.

Alternatively, a non-zero IVT can occur even when the distribution of variables is balanced with respect to global variable tightness, when agents have an imperfect awareness of constraints on variables they can assign. In this case, variables may not appear to be tight in the agent's local view when globally they are. This situation, which we call an *information-based* imbalance, is a violation of Condition 1. It is caused by the difference between the real and apparent variable tightness texture measures, or between the objective and subjective views. An information-based IVT, in contrast to an inherent one, may be corrected by providing more information to the assigning agent.

2.3 Example

Agent	Order	Deadline	Activities With Resource and Time Requirements
α	1	16	A_1 (R2 for 4) $\rightarrow$ A_2 (R1 for 5) $\rightarrow$ A_3 (R2 for 2)
α	2	8	A_4 (R1 for 2) $\rightarrow$ A_5 (R4 for 2)
β	3	16	A_6 (R4 for 4) $\rightarrow$ A_7 (R2 for 3) $\rightarrow$ A_8 (R3 for 2) $\rightarrow$ A_9 (R1 for 2)
β	4	16	A_{10} (R1 for 4) $\rightarrow$ A_{11} (R1 for 3) $\rightarrow$ A_{12} (R2 for 4)

Figure 1. Example of a scheduling problem.

Figure 1 shows a scheduling problem for two agents, each with two orders consisting of a sequence of activities with resource and time requirements shown. All orders are released at time 0 and must be completed by their deadline. For each activity, a domain of possible start times is constructed based on precedence relationships within the containing order. For example, Activity 2 cannot begin before Activity

1 has finished, nor can it begin later than the Order 1 deadline minus its own duration plus that of Activity 3. Thus the domain for Activity 2 is {4, 5, 6, 7, 8, 9}. Problem solving consists of selecting an activity to assign and a possible start time from the activity's domain, until a consistent set of start times has been found.

```
Activity   10   10   10   10   4    4    11   11   11   2    2    2    2    2    9    9
R1:      |----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|

Activity    1    1    1    1    7    7    7              12   12   12   12             3    3
R2:      |----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|

Activity                                       8    8
R3:      |----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|

Activity    6    6    6    6              5    5
R4:      |----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|

Time     0    1    2    3    4    5    6    7    8    9    10   11   12   13   14   15   16
```

Figure 2. One solution to the example scheduling problem.

There is a great deal more flexibility in the scheduling of some activities than in others for this problem. To envision this, consider Figure 2, which shows one of the 512 consistent schedules. Three of the five activities requiring Resource R1, which is required for a total of 16 time units, have only one possible time at which they can feasibly be scheduled (Activities 11, 2, and 9), while each of the other two has two possible times (Activities 10 and 4, which can be scheduled as shown or switched with each other). No other values in the domains of these activities are ever included in solutions. On the other hand, some activities have considerably more flexibility: Activity 8 can be scheduled anytime between times 7 and 12, and Activity 5 can be scheduled anytime between times 2 (if Activities 10 and 4 are switched) and 15. In fact, every value in the domains of these two activities is a possible solution value. To minimize the amount of backtracking required to reach a consistent solution, start times should be assigned early in problem solving for those activities that are likely to initiate backtracking, and later for those less likely to do so.

Table 1 shows computation of exact, global variable tightness measures for the 12 activities. The variable tightness for each activity is calculated as the percentage of "deconstrained solutions" (consistent solutions to the problem excluding the activity in question) for which no consistent assignment for that activity can be found. As would be expected, variable tightness is very high, over 90%, for Activities 2, 4, 10, and 11, four of the five activities that require Resource R1. But not all activities that require a resource with little flexibility are tight: variable tightness for Activity 9, which also requires Resource R1, is 0.00. Thus if a solution to the entire problem excluding Activity 9 is found, then a consistent assignment for Activity 9 always remains. This is because, since Activity 2 must be scheduled no later than time 9 in order for Activity 3 to finish before the Order 1 deadline (Figure 2), Resource R1 will always be available at time 14, which is fine for Activity 9. The variable tightness measures for Activities 1 and 8 are not surprisingly 0.00, and others fall somewhere in between. Twelve different variable orderings are possible based on these variable tightness measures.

		Centralized View			Agent α: Local View		
Activity	Resource	Deconstr. Solutions	Infeasible Solutions	Variable Tightness	Deconstr. Solutions	Infeasible Solutions	Variable Tightness
1	R2	217	0	0.00	292	0	0.00
2	R1	26528	26016	0.98	540	230	0.43
3	R2	1404	892	0.64	310	0	0.00
4	R1	7848	7336	0.93	280	0	0.00
5	R4	344	24	0.07	258	0	0.00
		Centralized View			Agent β: Local View		
6	R4	408	136	0.33	2814	0	0.00
7	R2	804	525	0.65	5758	2944	0.51
8	R3	126	0	0.00	3532	1096	0.31
9	R1	512	0	0.00	2548	0	0.00
10	R1	18498	17986	0.97	2114	0	0.00
11	R1	20359	19847	0.97	2931	817	0.28
12	R2	482	172	0.36	2527	0	0.00

Table 1. Centralized and local views of variable tightness.

If variable tightness measures are used for parallel problem solving by two asynchronous agents, then either an imbalance in variable tightness or asynchrony of the agents can cause tighter variables to be assigned before looser ones. With respect to asynchrony, if, in our example, Agent β makes its first assignment before Agent α, then either Activity 10 or Activity 11 will be assigned before Activity 2. If agents carry out assignments at approximately the same rate, however, and if variables are divided more or less evenly among agents with respect to variable tightness, then deviations from the centralized view variable ordering due to asynchrony alone should be relatively minor. If variables are not evenly divided with respect to variable tightness, however, then large deviations can occur. In this example, for the first five (unordered) *pairs* of assignments carried out asynchronously by Agents α and β, we have (2 and 10/11) - (4 and 10/11) - (3 and 7) - (5 and 12) - (1 and 6), which will be followed by Activities 8 and 9, in either order, by Agent β. Note that Activity 5 (variable tightness = 0.07), is going to be assigned before Activity 12 (variable tightness = 0.33) half the time, and before Activity 6 (variable tightness = 0.33) all the time. This is an *inherent* imbalance in variable tightness, caused by the distribution of activities among the agents rather than by lack of information.

Even if variables are distributed evenly among the agents with respect to global variable tightness, locally-computed variable tightness measures can be quite different than global measures, such that an *information-based* IVT can arise. Table 1 also shows variable tightness measures for activities belonging to each agent based on local views, that is, only the two orders that they know about. The relative ranking of activities has changed compared to the centralized variable tightness ordering. Activities 4 and 10, which should be among the first four activities assigned, are as likely to be assigned last based on local variable tightness measures as the activities that belong there based on global variable tightness.

For this problem, variance in variable tightness is 0.217 and 0.168 for Agents α and β, respectively, giving a variance ratio $F_s = 1.29$, $p(F_s) = 0.89$, which is not significant. A t-test for equality of means resulted in $t_s = 0.403$, $p(t_s) = 0.71$. Thus IVT = 1 - $p(t_s) = 0.29$.

2.4 Evaluation

In order to evaluate the IVT texture as a predictor of problem solving efficiency in a distributed system, we developed a simulation of distributed problem solving in the domain of scheduling, modeled closely after Sycara et al.'s distributed constrained heuristic search system [Sycara et al. 1991], with a few exceptions. One difference is that we have not included communication delays; rather, at this point we are looking at the best case scenario of instantaneous communication, via a shared data structure. In the future we will study the relationship between problem structure, communication strategies, and problem solving efficiency, but for looking specifically at the effect of IVT on problem solving efficiency, communication delays would be a confounding factor. Second, we have included in our simulation system the ability to do variable ordering in any of four ways: by Sycara et al.'s method of information sharing; randomly; or in order of either the local or the global variable tightness measures. Similarly, for value ordering we can use either Sycara et al.'s method or random value ordering.

In Sycara et al.'s system, it is sometimes the case that one or more agents can find no feasible solution to their part of the problem while other agents have solved their parts easily, when in fact a feasible solution to the problem does exist. Because we wanted to measure the effort involved in finding a solution using different scheduling orders, we could not allow a poor variable ordering to result in "no solution", so we added an inter-agent backtracking protocol that is guaranteed to find a solution to problems for which a feasible solution exists. Our inter-agent backtracking protocol, though developed independently, is similar to that described in [Yokoo et al. 1992]: it avoids infinite processing loops by using a total order relationship among agents. Our protocol is different in that agents are not restricted to one shared variable and, since communication is via a shared data structure, agents do not have to know which agents they share constraints with.

2.4.1 Experiment 1: Inherent IVT and Problem Solving Efficiency

Our first experiment was designed to evaluate whether inherent IVT is a good predictor of problem solving efficiency. When inherent IVT is low, differences in variable ordering between distributed and centralized systems are due mostly to agent asynchrony. If the effect of agent asynchrony is relatively minor, then when inherent imbalance is low, distributed system performance should come close to the ideal of problem solving in order of exact global variable tightness. Thus we predicted a negative correlation between IVT and number of assignments required (including ones that were backed off) to find an overall feasible schedule. Note that we were *not* evaluating this method as a viable distributed problem solving strategy, since computing the exact global variable tightness measures requires solving the problem repeatedly, from a centralized point of view. Rather, our goal is to understand the underlying relationship between the distribution of variables among agents with respect to global variable tightness and problem solving efficiency.

We simulated distributed problem solving for 18 scheduling problems involving two agents doing variable ordering based on global variable tightness measures. Each agent was responsible for scheduling one to three orders, each consisting of one to six activities. All 18 problems had a total of 12 activities to be scheduled. Agents worked asynchronously and in parallel, choosing activities to schedule in order of their global variable tightness measures, which were pre-computed and stored. Once an activity was chosen, agents selected a possible scheduling time at random. Agents used exact global variable tightness measures for variable ordering so that they had perfect information on variable tightness; thus the factors affecting distributed problem solving efficiency in this experiment were problem distribution among the agents (inherent IVT), agent asynchrony, and the random effect of random value ordering. Our sample size for this experiment was 1000 trials for each of the 18 problems.

We collected as raw data the number of reservations (variable assignments) made. However, comparing strategy efficiencies using reservations made was confounded by the fact that some problems were inherently more difficult than others: the mean number of reservations required to reach a consistent solution using a trial and error approach (random variable and value ordering, 2000 trials) ranged from 32 to 2594 for the 18 problems (Table 2). Therefore we measured the distributed system's performance as a relative gain measure: a ratio of the distributed system's improvement over a baseline of random variable and value ordering and the centralized system's improvement over the same baseline. We can write this as:

$$\text{relative gain} = \frac{\overline{R}_{CRR} - \overline{R}_{DGR}}{\overline{R}_{CRR} - \overline{R}_{CGR}}$$

where the subscripts are as follows: first position indicates Distributed or Centralized system; second position indicates variable ordering based on Global variable tightness or Random; and third position indicates Random value ordering. Differences between the two gain measures will be due to differences in variable ordering caused by agent asynchrony and uneven distribution of activities among agents with respect to variable tightness. This ratio would generally run from 0 to 1, although values of below 0 (distributed performance worse than random) or above 1 (distributed performance better than centralized) are possible.

Table 2 shows mean number of reservations required to reach a consistent solution for the centralized and distributed systems using variable ordering based on global variable tightness measures. The data for centralized problem solving using random variable ordering are included as a baseline of problem difficulty. Medians, maximums and standard deviations are included as representative of typical performance, worst performance, and variation in performance. The best possible performance, finding a solution in 12 reservations, was observed at least once for every problem for all strategies. The last column shows relative gain over baseline for the distributed versus the centralized system. As an example computation, Problem 1 gain ratio $= \frac{57-17}{57-13} = \frac{40}{44} = 0.91$.

Problem	Centralized/ Random	Centralized/ Global VT	Distributed/ Global VT	Gain Ratio
1	57 (19, 1258, 107)	13 (13, 13, 1)	17 (18, 24, 4)	0.91
2	937 (183, 27413, 2304)	24 (24, 101, 7)	164 (68, 801, 180)	0.85
3	435 (23, 25934, 1790)	15 (14, 21, 2)	26 (14, 393, 52)	0.97
4	779 (15, 126239, 5449)	13 (13, 20, 2)	17 (17, 30, 5)	0.99
5	33 (13, 3478, 160)	12 (12, 12, 0)	13 (12, 21, 2)	0.95
6	2594 (638, 110638, 6421)	83 (84, 181, 41)	1394 (1286, 4676, 1387)	0.48
7	77 (16, 6112, 293)	14 (14, 16, 1)	25 (19, 61, 14)	0.83
8	216 (24, 16648, 814)	16 (16, 24, 4)	25 (20, 85, 14)	0.96
9	368 (16, 71417, 3143)	15 (13, 60, 7)	15 (13, 59, 7)	1.00
10	95 (16, 6740, 450)	13 (13, 16, 1)	17 (17, 27, 4)	0.95
11	34 (15, 1378, 84)	13 (13, 15, 1)	16 (16, 27, 4)	0.86
12	159 (17, 13031, 717)	12 (12, 12, 0)	13 (12, 22, 2)	0.99
13	344 (121, 5465, 588)	23 (23, 35, 9)	96 (81, 336, 71)	0.77
14	329 (42, 12829, 946)	18 (18, 29, 4)	17 (14, 29, 5)	1.00
15	32 (20, 570, 34)	22 (26, 31, 7)	33 (25, 82, 18)	-0.10
16	44 (25, 653, 54)	16 (16, 22, 3)	44 (26, 112, 32)	0.00
17	186 (17, 16161, 870)	14 (13, 18, 2)	22 (15, 67, 17)	0.95
18	1240 (161, 55524, 3632)	20 (16, 44, 10)	2316 (2011, 6197, 2204)	-0.88

Table 2. Results of Experiment 1: Mean (median, maximum, standard deviation) number of reservations to find a consistent solution for centralized and distributed problem solving using variable ordering based on global variable tightness measures, and the gain over centralized random variable ordering for distributed versus centralized problem solving.

Gain ratios are plotted against IVT in Figure 3. Efficiency does not correlate well with IVT. The trend toward a positive correlation between the problem solving efficiency gain measure and IVT is not significant; a linear regression explains only 11% of the variance and has a probability of 0.178. Thus while there were dramatic differences in observed efficiencies, they did not seem to relate to this statistical measure of imbalance in the distribution of variables among the agents with respect to variable tightness.

2.4.2 Experiment 2: Information-Based IVT and Problem Solving Efficiency

Though the results of Experiment 1 were negative, we thought it worthwhile to test for a relationship between information-based IVT and problem solving efficiency. Experiment 2 was identical to Experiment 1 except that variable ordering in the distributed system was based on local rather than global variable

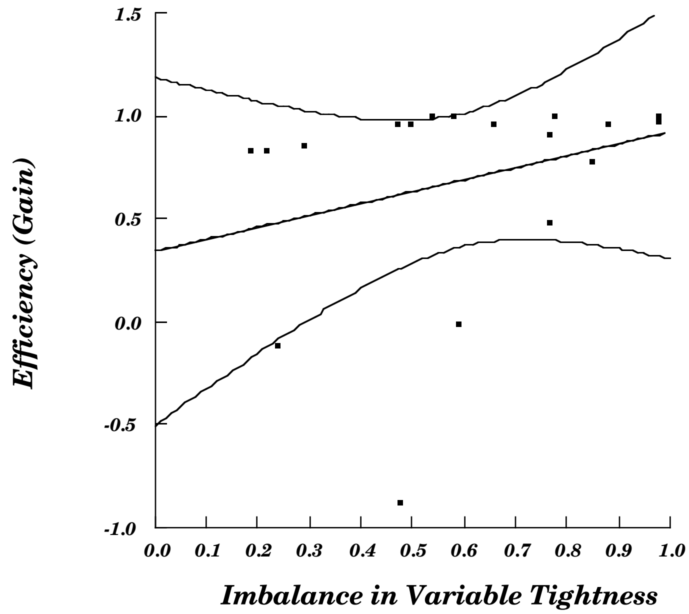

Figure 3. Relationship between IVT and distributed problem solving efficiency using variable ordering based on global variable tightness, 95% confidence intervals.

tightness measures. Thus in Experiment 2, the difference in problem solving efficiency from the centralized system is due to agent asynchrony and distribution of variables among agents, as before, and in addition,

Problem	Distributed Local VT		Gain Ratio
1	28	(20, 94, 17)	0.66
2	2512	(2584, 5982, 1922)	-1.73
3	17	(14, 207, 16)	0.99
4	98	(23, 1822, 187)	0.90
5	17	(15, 39, 6)	0.77
6	1394	(1497, 4521, 1401)	0.48
7	28	(17, 307, 35)	0.77
8	45	(42, 118, 31)	0.85
9	15	(13, 59, 7)	1.00
10	18	(17, 31, 4)	0.94
11	18	(18, 34, 4)	0.73
12	14	(12, 20, 2)	0.99
13	178	(152, 559, 130)	0.52
14	16	(15, 29, 4)	1.00
15	52	(52, 101, 22)	-2.12
16	45	(31, 172, 37)	-0.04
17	18	(15, 53, 9)	0.98
18	786	(65, 5236, 1323)	0.37

Table 3. Results of Experiment 2: Mean (median, maximum, standard deviation) reservations to find a consistent solution for distributed problem solving using variable ordering based on local measures of variable tightness, and gain over centralized problem solving using random variable relative to centralized problem solving.

loss of constraint information due to local, subjective views. The relative gain measure used in Experiment 1 was used again in Experiment 2 to assess problem solving efficiency. Sample size was 1000 trials.

Not surprisingly, there was again no significant relationship between problem solving efficiency and IVT. Table 3 shows mean number of reservations to reach a consistent solution in the distributed system using variable ordering based on local variable tightness, and the ratio of distributed gain over centralized gain, relative to baseline of centralized problem solving using random variable ordering. Minimum number of reservations observed for all problems was 12, the minimum possible.

Figure 4 shows the gain measure as a function of IVT. Just as for distributed problem solving using global variable tightness measures for variable ordering, while there is a trend toward a positive relationship, a regression analysis is not significant ($p = 0.276$), and only 7.4% of the variance is explained.

2.4.3 Discussion

In both Experiments, the same general effect was tested, namely the effect of agent asynchrony plus distribution of variables among agents with respect to variable tightness on problem solving efficiency. In a distributed system, even when using global measures of variable tightness for value ordering, deviations from the variable ordering that would obtain in a centralized system can occur because of agent asynchrony and an uneven distribution, with respect to variable tightness, of variables among agents. The more unevenly variables are distributed with respect to variable tightness, the further the resulting (overall) variable ordering will be from a centralized system doing variable ordering based on global variable tightness. The result of Experiment 1, that problem solving efficiency is not negatively correlated with IVT, shows that even small deviations can have a major effect on problem solving efficiency, and

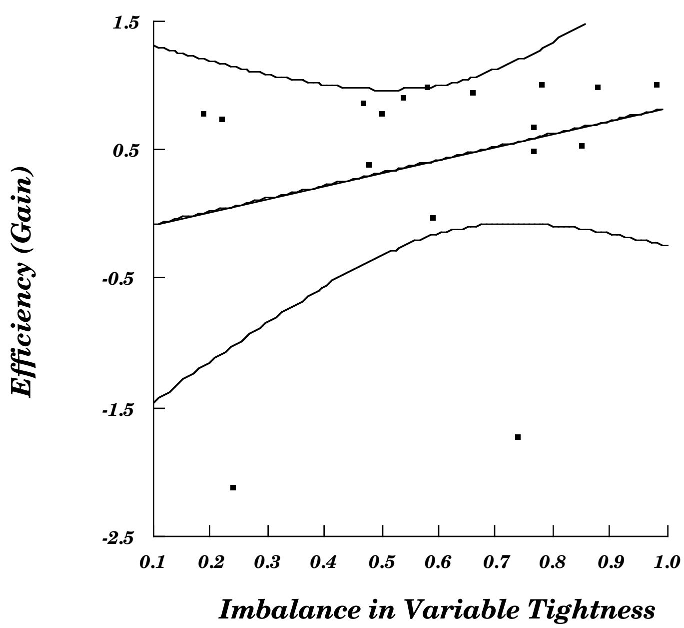

Figure 4. Relationship between IVT and distributed problem solving efficiency using variable ordering based on local variable tightness, with 95% confidence intervals, $p = 0.276$, $r^2 = 0.074$.

conversely, large deviations sometimes make little difference.

When using local variable tightness measures, deviations from global variable tightness ordering are likely to be even larger, since in addition to the effect of variable distribution among agents, there is the added effect of using subjective information, namely local variable tightness measures that may be based on incomplete information and may differ substantially from global measures. Experiment 2 tested the effect of both information-based and inherent imbalances. The results of Experiment 2 are consistent with those of Experiment 1: that small differences in variable ordering can sometimes have a large effect, and that large differences can sometimes matter very little, such that a statistical measure of the distribution of variables among agents is simply not predictive.

We know, however, that the order in which variables are assigned matters greatly. This is demonstrated empirically by the large differences in number of reservations needed to reach a consistent solution for problem solving using random variable ordering versus variable ordering by global variable tightness (Table 2, second and third columns). There must, then, be particular precedence relations within a variable order based on global variable tightness that are significant, while others are not. If these relations are maintained in a variable order that otherwise differs markedly from that based on global variable tightness, problem solving efficiency would remain high. Conversely, a variable order that was identical to that based on global variable tightness except for one reversal might result in extremely poor efficiency. This idea is explored in Section 3.

3 Precedence Relationships Among Variables

Consider the meaning of variable tightness: a tight variable is one that, when assigned last, is unlikely to have any value remaining in its domain that is consistent with the rest of the solution; that is, there is a high probability that assignments to other variables will have eliminated all of its domain values as viable possibilities. The earlier tight variables are assigned, the higher the probability that feasible solution values will not already have been eliminated by assignments to other variables. Probabilistically, over all possible solution paths, backtracking is minimized by assigning variables in order of variable tightness.

In any given constraint satisfaction problem, however, not every variable is necessarily interdependent with every other. Thus it is not necessarily the case for each variable that every other variable of lower variable tightness will remove feasible solution values from its domain; rather, some variables do, while others do not. Although a given variable may be very tight, it is really only important that its assignment precede assignments to those variables that reduce its domain by values that could appear in a solution. Its order of assignment relative to other variables that do not affect its domain, regardless of their variable tightness, is irrelevant to problem solving efficiency. We refer to constraints among variables that remove solution values from domains as *negative*. Negative constraints are the underlying cause of variable tightness.

Additionally, constraints among variables can be *positive*: variables may remove non-solution values from the domains of other variables, with a positive effect on problem solving efficiency. Since we are looking for important precedence relations between variables, we consider that the precedence of one variable before another may be an important one if the former removes non-solution values from the domain of the latter.

We define *significant precedence relations* to be those that, when honored, improve problem solving efficiency over random variable ordering, and when violated, have a detrimental effect on problem solving efficiency. We conjecture that significant precedence relations can be identified by determining those pairs of variables in which one member can remove either solution or non-solution values from the domain of the other. Specifically, we hypothesize that if any assignment to a variable w can remove one or more values that ever appears in a solution from the domain of a variable v, then ***v before w*** is a significant precedence relation, and similarly, if any assignment to a variable v can remove one or more values that never appears in a solution from the domain of a variable w, then ***v before w*** is a significant precedence relation. Furthermore, the contribution of significant precedence relations to problem solving should be quantifiable, based on the frequency with which solution and non-solution values are removed over all solution paths.

3.1 Experiment 3: Significant Precedence Relations

In order to test the significance of precedence relations generated according to these definitions, we generated all precedence relations fitting the definitions above for Problem 1 from the 18 scheduling problems used in Experiments 1 and 2, and evaluated the significance of each using a controlled, paired analysis. When negative or positive constraints were bi-directional, or when negative and positive constraints suggested precedence constraints in opposite directions, a precedence relation was created in the direction in which the constraints were stronger. To test each relation ***A before B***, we ran paired trials matched for variable ordering except for swapping activities A and B, which were adjacent. The same random state was used at the start of both members of a paired trial. Problem 1 was a fairly easy problem (57 reservations on average using random variable ordering).

To test the significance of individual precedence relations, we could not simply generate a large number of trials using random variable and value ordering and, for each precedence relation, compare the numbers of reservations made in those runs in which the precedence relation was met with those runs in which it was not. This is not really indicative of the significance of the precedence relation in question for the following reason: suppose we are looking at the precedence relation ***A before B***, and we compare those runs in which A precedes B with those in which B precedes A. But note that in the set of runs in which A precedes B, A is assigned on average earlier than it is in the set in which B precedes A. Not only B, but *every* variable is more likely to follow than precede A in the ***A before B*** set, because A is assigned on average earlier in that set than in the other set.

This problem was remedied by doing paired trials, in which *A* and *B* were adjacent to each other but with order reversed in the two trials. In these matched trials, the variables that preceded *A* in one trial were identical to those that preceded *A* in the other, except for *B*. Any difference in the paired trials could *only* be due to the relative order of *A* and *B*, and to chance due to the effect of different (randomly generated) value orderings. In order to reduce the effect of chance, we began both members of a pair of trials with the same random state, so that the trials were completely identical up to the point at which the first of *A* or *B* was assigned. Beyond that point, there was, of course, no way to make the trials identical, so we used statistical analyses to pick the effect of the precedence relation out of the noise.

For Problem 1, with a sample size of 500 paired trials for each of 11 precedence relations predicted to be significant, six precedence relations had a highly significant beneficial effect on the number of reservations made (Wilcoxon 2-sample test, $p \leq .01$), four had no significant effect ($p > .05$), and one had a highly significant detrimental effect ($p < .001$).

The precedence relation with the negative effect, ***Activity 4 before Activity 5***, was a puzzle: while the precedence relation was in the opposite direction than one would predict based on variable tightness (Activities 4 and 5 had variable tightness measures of 0.00 and 0.99, respectively), Activity 4 acted only as a positive constraint on Activity 5, and Activity 5 did not constrain Activity 4 at all. (Constraint relationships can be asymmetrical when, due to other constraints, one activity is prevented from taking on values that constrain the other.) Activity 5 had a domain of three values, only one of which was ever included in any consistent solution. In 10 of the 15 different consistent solutions to the problem, Activity 4's assignment blocked one of the two infeasible assignments for Activity 5, and in 14 of the 15 consistent solutions, it blocked both; these were positive constraints. In addition, no value in Activity 4's domain was inconsistent with the single solution value in Activity 5's domain, that is, there were no negative constraints. Therefore, how could it possibly be detrimental for Activity 4 to precede Activity 5?

Since Activity 5 was a very tight variable, the later it came in the ordering of assignments, the more likely it was that it would not have a consistent assignment remaining in its domain, and the longer the paths involved in backtracking repeatedly to find a combination of assignments for earlier variables that would allow it a consistent assignment. Since Activity 4 was not one of the variables that was negatively constraining Activity 5, its preceding Activity 5 only increased the length of the path from Activity 5 back to the activity or activities that were negatively constraining it. Thus the precedence relation ***Activity 4 before Activity 5*** was negative *in a context in which some activities that should have followed Activity 5 did not necessarily do so*. When we repeated the paired trials for this precedence relation in a context in which all four of the other precedence relations involving Activities 5 and 4 were met, the precedence relation Activity 4 before Activity 5 became highly significant ($p < .001$) in the right direction.

What about the four non-significant precedence relations? Evaluating them in context as above, they still showed no significant effect, and close examination of their negative and positive constraint effects over all solution paths did not suggest they should be weaker relations. A possible explanation is that the importance of these particular precedence relations is superseded by other precedence relations, and that once those are met, these precedence relations have no additional effect. As evidence for this, we observed that in the vast majority of paired runs, the *exact* same number of reservations were made for the precedence met and precedence violated members of the pair for these four relations (range was 462 - 485 out of 500 paired trials), meaning that there was no effect at all. This degree of exact matching was not characteristic of the paired trials for these relations not involving "context"; in those, the number of trials in which the pair members matched exactly with respect to number of reservations varied from 366 to 439 out of 500 paired trials: more than for the precedence relations that showed significant effects, but not as high as "in context". In the paired trials in which these precedence relations were tested without controlling their context, it is possible that the superseding precedences happened to be met often enough to wash out the effect of the precedences being tested.

From this analysis it was clear that the contribution of individual precedence relations can sometimes depends on context. We next decided to evaluate whether sets of predicted precedence relations worked well together as a whole. We generated precedence relations using a method combining the idea of domain reductions with that of variable tightness: we found all pairs of variables that reduced each others domains at all and generated a precedence relation for each pair in the direction of decreasing variable tightness. For each problem we looked at efficiency of problem solving using random variable ordering within the constraint of meeting the set of precedence relations, a strategy we call random-constrained, or rand-con. Table 4 shows problem solving efficiency for each problem for variable ordering done (1) randomly, (2) based on global variable tightness measures, and (3) using the rand-con strategy. Sample sizes were 2000 trials for random ordering and 1000 for the other methods.

The column "# Orderings" shows the number of different variable orderings possible using each variable ordering method. For 12 activities ordered randomly, there are 12! or 479,001,600 possible orders. There are many fewer possible orderings based on global variable tightness: between 4 and 1440, varying with the problem. Using the rand-con method, there remain thousands of possible orderings for each of these 18 problems. The difference in problem solving efficiency between using the rand-con method and using global variable tightness varies between none and substantial. For seven of the 18 problems (1, 4, 5, 11, 12, 15, and 17), the rand-con method performed within 10% of variable ordering using global variable tightness. For the others, however, although the rand-con method was substantially better than random ordering, it was sufficiently poor relative to global variable tightness variable ordering that all precedence information important to performance must not have been captured, or alternatively, that precedence relations within the generated sets exhibited conflicts. Careful examination of the order of variable assignments in runs that exhibited large amounts of backtracking showed that there were no binary precedence orderings that *always* occurred in inefficient runs versus efficient ones, or vice versa, although there were some that were observed *infrequently* in one or the other case.

Problem	Method	# Orderings	Mean	Maximum	Median	Std. Dev.
1	random	479,001,600	57.2	1,258	19	107.4
	global VT	720	13.0	14	13	0.8
	rand-con	97,545	12.7	14	12	0.8
2	random	479,001,600	936.8	27,413	182.5	2303.8
	global VT	12	23.6	101	24	6.8
	rand-con	34,477	159.9	2903	53	305.6
3	random	479,001,600	435.2	25,934	23	1789.7
	global-VT	240	14.7	21	14	2.1
	rand-con	1,663,200	18.9	1180	15	45.0
4	random	479,001,600	779.5	126,239	15	5448.9
	global VT	6	13.4	20	13	1.6
	rand-con	54,120	13.5	20	13	1.6
5	random	479,001,600	33.0	3,478	13	160.0
	global VT	720	12.0	12	12	0.0
	rand-con	475,200	12.0	12	12	0.0
6	random	479,001,600	2593.7	110,638	637.5	6421.0
	global VT	288	83.1	181	83.5	41.1
	rand-con	6036	119.9	828	99	107.1
7	random	479,001,600	77.4	6112	16	295.5
	global VT	24	13.7	16	14	1.1
	rand-con	26,433	18.3	66	14	9.7
8	random	479,001,600	215.8	16,648	24	814.4
	global VT	120	15.7	24	16	3.6
	rand-con	29,304	62.5	4089	13	273.1
9	random	479,001,600	368.1	71,417	16	342.6
	global VT	48	15.1	60	13	6.8
	rand-con	285,560	22.3	1169	13	72.2
10	random	479,001,600	94.8	6740	16	450.4
	global VT	6	13.4	16	13	1.2
	rand-con	21,450	19.2	125	13	19.3
11	random	479,001,600	33.5	1378	15	83.8
	global VT	24	12.9	15	13	0.8
	rand-con	171,640	12.9	15	13	0.9
12	random	479,001,600	159.4	13,031	17	717.4
	global VT	1,440	12.0	12	12	0.0
	rand-con	38,720	12.0	12	12	0.0
13	random	479,001,600	344.4	5465	120.5	588.3
	global VT	120	23.1	35	23	8.6
	rand-con	16,524	27.0	58	27	12.6
14	random	479,001,600	329.4	12,829	42	945.7
	global VT	120	17.9	29	18	3.6
	rand-con	121,528	157.9	1404	48.5	221.2
15	random	479,001,600	31.6	570	20	34.3
	global VT	24	21.9	31	26	6.9
	rand-con	172,620	17.4	50	16	5.1
16	random	479,001,600	43.9	653	25	53.7
	global VT	4	15.8	22	16	2.5
	rand-con	231,075	21.4	95	16	12.8
17	random	479,001,600	185.9	16,161	17	869.8
	global VT	24	13.8	18	13	1.6
	rand-con	4,869	14.6	27	14	3.0
18	random	479,001,600	1240.1	55,524	160.5	3631.6
	global VT	96	20.3	44	16	9.6
	rand-con	35,412	284.0	4379	54	555.1

Table 4. Relative effectiveness of the rand-con strategy (random variable ordering constrained by a set of precedence relations based on positive and negative constraints among the variables).

While the rand-con method is not fully as good as variable ordering by global variable tightness in a centralized system, in a distributed system the rand-con method is generally better in efficiency than variable ordering by either local or global variable tightness. Although problem solving might proceed faster in a distributed system because of parallel problem solving, efficiency measures for rand-con obtained in a centralized setting measured in terms of number of reservations would be unaffected by moving to a distributed system. Thus we can compare the numbers obtained above with the numbers of reservations made in a distributed system using global variable tightness for variable ordering (Table 2) or in a distributed system using local variable tightness (Table 3). Rand-con performs better than global variable tightness for 14 out of 18 problems, and better than local variable tightness for 13 out of 18. The overlap of three problems for which rand-con does not outperform either variable tightness method, Problems 8, 9, and 14, again suggests that not all of the important precedence information has been captured by the rand-con strategy.

To ensure that our rand-con results were not affected adversely by our inclusion of precedence relations based on a combination of domain reductions and variable tightness, we carefully computed precedence relations using the exact definitions of negative and positive domain constraints. In the case of bi-directional constraints, or where negative and positive constraints suggested opposite precedence relations, we chose based on the stronger constraints, ignoring variable tightness. For Problems 1 and 2, these sets of precedence relations were much worse, and we did not pursue this any further.

3.1 Discussion

The results of Experiment 3 and the follow-up experiments described suggest that considering binary precedence relations in isolation is not powerful enough to capture the important relationships among variables that can affect problem solving efficiency. We have seen that the effect of a particular precedence relation can depend additionally on context, meaning where in the order of assignments the precedence relation falls with respect to other precedence relations that are also important. This conclusion is supported by the observation that no binary order relationships occurred exclusively in trials that exhibited large amounts of backtracking versus trials that found solutions in a small number assignments; perhaps instead certain partial orders are detrimental in the context of other particular partial orders. In addition, we have found that using all precedence relations that should improve problem solving together as a set does not necessarily have the desired beneficial effect. It is possible that some precedence relations are in conflict with others, such that for maximal efficiency some subset should be adhered to, but not all, and that the effect of some precedence relations is superseded by others. Within the set of precedence relations for a problem, it is likely that some are more critical than others, such that when that information is available, they should be adhered to while others are ignored. However, in the case of incomplete information about relationships among precedence relations, it may be better to include the weaker precedence relations than to leave them out.

4 Summary: Implications and Future Directions

In this paper we defined a texture measure appropriate for distributed systems, called *Imbalance in Variable Tightness*, or IVT, as a measure of the unevenness in distribution of variables among agents with respect to variable tightness. In experiments to evaluate this texture, however, we found that its predictive ability was low, not because variable order is unimportant, but because as a statistical measure it was not able to capture the violation of or adherence to particular partial orders among variables that were important to problem solving efficiency. Closer examination of the importance of particular binary precedence relations showed that even that level of analysis was too crude to predict problem solving efficiency. Rather, the contribution of binary precedence relations to problem solving efficiency depended on context with respect to other variable assignments or other groups of variable assignments.

This work suggests that to understand the relationship among search space, problem solving strategies, and problem solving efficiency, it is important to consider not only first-order precedence relationships among variables, but second-order effects, that is, the effect of partial orders on each other, as well. We are in the process of categorizing these relationships and developing a terminology within which to describe and quantify their effect on problem solving efficiency. The observation that these higher-order relationships among variables can be important in determining problem solving efficiency has implications for task decomposition in distributed systems, for timing of communication of partial results, for interpretation of partial results, for when synchronization might be important, for information that could be exchanged that would speed up problem solving, and for choosing what activities may be contributory, and what activities may be futile, to pursue at any given time in problem solving.

5 Appendix: The 18 Scheduling Problems

Problem 1: 15 solutions IVT = 0.77

Agent	Order	Deadline	Activities With Resource and Time Requirements
α	1	19	A_1 (R1 for 2) → A_2 (R3 for 4) → A_3 (R3 for 3) → A_4 (R3 for 4) → A_5 (R2 for 2) → A_6 (R1 for 2)
β	2	19	A_7 (R3 for 2) → A_8 (R2 for 4) → A_9 (R1 for 2) → A_{10} (R1 for 3) → A_{11} (R2 for 4) → A_{12} (R2 for 2)

Problem 2: 512 solutions IVT = 0.74

Agent	Order	Deadline	Activities With Resource and Time Requirements
α	1	16	A_1(R2 for 4) → A_2 (R1 for 5) → A_3 (R2 for 2)
α	2	8	A_4 (R1 for 2) → A_5 (R4 for 2)
β	3	16	A_6 (R4 for 4) → A_7 (R2 for 3) → A_8 (R3 for 2) → A_9 (R1 for 2)
β	4	16	A_{10} (R1 for 4) → A_{11} (R1 for 3) → A_{12} (R2 for 4)

Problem 3: 2450 solutions IVT = 0.98

Agent	Order	Deadline	Activities With Resource and Time Requirements
α	1	8	A_1 (R2 for 2)
α	2	20	A_2 (R3 for 4) → A_3 (R4 for 3) → A_4 (R4 for 3) → A_5 (R2 for 3) → A_6 (R2 for 4)
β	3	16	A_7 (R1 for 4) → A_8 (R3 for 4) → A_9 (R3 for 4)
β	4	16	A_{10} (R1 for 4) → A_{11} (R3 for 4) → A_{12} (R1 for 2)

Problem 4: 25837 solutions IVT = 0.54

Agent	Order	Deadline	Activities With Resource and Time Requirements
α	1	8	A_1 (R2 for 2) → A_2 (R1 for 2)
α	2	16	A_3 (R2 for 2) → A_4 (R3 for 2) → A_5 (R3 for 5)
β	3	8	A_6 (R4 for 4) → A_7 (R2 for 2)
β	4	20	A_8 (R1 for 5) → A_9 (R1 for 2) → A_{10} (R4 for 3) → A_{11} (R2 for 2) → A_{12} (R2 for 4)

Problem 5: 2289 solutions IVT = 0.50

Agent	Order	Deadline	Activities With Resource and Time Requirements
α	1	8	A_1 (R3 for 4) → A_2 (R2 for 2)
α	2	20	A_3 (R4 for 4) → A_4 (R3 for 4) → A_5 (R3 for 4) → A_6 (R3 for 2) → A_7 (R1 for 2) → A_8 (R4 for 2)
β	3	8	A_9 (R2 for 2) → A_{10} (R1 for 3)
β	4	8	A_{11} (R1 for 2) → A_{12} (R1 for 2)

Problem 6: 91 solutions IVT = 0.77

Agent	Order	Deadline	Activities With Resource and Time Requirements
α	1	16	A_1 (R4 for 4) → A_2 (R3 for 2) → A_3 (R3 for 4)
α	2	16	A_4 (R2 for 4) → A_5 (R3 for 2) → A_6 (R3 for 2)
β	3	16	A_7 (R3 for 4) → A_8 (R4 for 5)
β	4	16	A_9 (R4 for 5) → A_{10} (R3 for 2) → A_{11} (R1 for 2) → A_{12} (R2 for 3)

Problem 7: 955 solutions IVT = 0.19

Agent	Order	Deadline	Activities With Resource and Time Requirements
α	1	16	A_1 (R3 for 5) → A_2 (R1 for 4) → A_3 (R2 for 3)
α	2	16	A_4 (R1 for 3) → A_5 (R3 for 5) Agent 2
β	3	8	A_6 (R1 for 2) → A_7 (R2 for 4)
β	4	20	A_8 (R4 for 3) → A_9 (R4 for 5) → A_{10} (R4 for 2) → A_{11} (R3 for 5) → A_{12} (R2 for 4)

Problem 8: 7028 solutions IVT = 0.47

Agent	Order	Deadline	Activities With Resource and Time Requirements
α	1	16	A_1 (R2 for 2) → A_2 (R1 for 3) → A_3 (R4 for 5)
α	2	16	A_4 (R3 for 5) → A_5 (R3 for 5)
β	3	20	A_6 (R4 for 2) → A_7 (R2 for 3) → A_8 (R1 for 3) → A_9 (R4 for 4) → A_{10} (R4 for 4)
β	4	16	A_{11} (R2 for 5) → A_{12} (R1 for 5)

Problem 9: 24820 solutions IVT = 0.78

Agent	Order	Deadline	Activities With Resource and Time Requirements
α	1	16	A_1 (R2 for 5) → A_2 (R1 for 3)
α	2	16	A_3 (R2 for 3) → A_4 (R2 for 5) → A_5 (R4 for 2)
β	3	16	A_6 (R4 for 3) → A_7 (R4 for 2) → A_8 (R4 for 4) → A_9 (R1 for 3)
β	4	8	A_{10} (R1 for 2) → A_{11} (R3 for 2) → A_{12} (R3 for 2)

Problem 10: 2016 solutions IVT = 0.66

Agent	Order	Deadline	Activities With Resource and Time Requirements
α	1	8	A_1 (R1 for 2) → A_2 (R1 for 2)
α	2	8	A_3 (R1 for 2) → A_4 (R4 for 2)
β	3	16	A_5 (R2 for 2) → A_6 (R4 for 4) → A_7 (R3 for 4) → A_8 (R4 for 4)
β	4	20	A_9 (R2 for 5) → A_{10} (R2 for 5) → A_{11} (R1 for 4) → A_{12} (R2 for 2)

Problem 11: 190 solutions IVT = 0.22

Agent	Order	Deadline	Activities With Resource and Time Requirements
α	1	20	A_1 (R4 for 5) → A_2 (R4 for 3) → A_3 (R2 for 4) → A_4 (R4 for 4) → A_5 (R4 for 2) → A_6 (R2 for 1)
α	2	8	A_7 (R3 for 2) → A_8 (R3 for 4)
β	3	8	A_9 (R2 for 2) → A_{10} (R3 for 2)
β	4	16	A_{11} (R2 for 5) → A_{12} (R1 for 5)

Problem 12: 832 solutions IVT = 0.58

Agent	Order	Deadline	Activities With Resource and Time Requirements
α	1	8	A_1 (R3 for 3) → A_2 (R4 for 2)
α	2	8	A_3 (R4 for 2) → A_4 (R1 for 3) → A_5 (R4 for 2)
β	3	16	A_6 (R1 for 2) → A_7 (R1 for 3) → A_8 (R3 for 5) → A_9 (R2 for 2)
β	4	16	A_{10} (R2 for 5) → A_{11} (R4 for 2) → A_{12} (R1 for 4)

Problem 13: 265 solutions IVT = 0.85

Agent	Order	Deadline	Activities With Resource and Time Requirements
α	1	8	A_1 (R3 for 5)
α	2	16	A_2 (R1 for 4) → A_3 (R3 for 2) → A_4 (R4 for 3) → A_5 (R1 for 2)
β	3	16	A_6 (R2 for 3) → A_7 (R3 for 3) → A_8 (R4 for 4) → A_9 (R4 for 2)
β	4	16	A_{10} (R1 for 5) → A_{11} (R3 for 2) → A_{12} (R2 for 3)

Problem 14: 366 solutions IVT = 0.98

Agent	Order	Deadline	Activities With Resource and Time Requirements
α	1	16	A_1 (R2 for 3) → A_2 (R1 for 5) → A_3 (R3 for 3)
α	2	16	A_4 (R4 for 3) → A_5 (R3 for 4) → A_6 (R4 for 2) → A_7 (R2 for 5)
β	3	16	A_8 (R1 for 3) → A_9 (R4 for 4) → A_{10} (R4 for 3)
β	4	8	A_{11} (R2 for 3) → A_{12} (R4 for 3)

Problem 15: 2 solutions IVT = 0.24

Agent	Order	Deadline	Activities With Resource and Time Requirements
α	1	8	A_1 (R3 for 3) → A_2 (R1 for 3)
α	2	8	A_3 (R2 for 2) → A_4 (R4 for 4)
β	3	8	A_5 (R2 for 2) → A_6 (R1 for 2) → A_7 (R2 for 3)
β	4	20	A_8 (R3 for 4) → A_9 (R1 for 4) → A_{10} (R3 for 3) → A_{11} (R4 for 2) → A_{12} (R2 for 4)

Problem 16: 26 solutions IVT = 0.59

Agent	Order	Deadline	Activities With Resource and Time Requirements
α	1	20	A_1 (R1 for 3) → A_2 (R3 for 2) → A_3 (R1 for 3) → A_4 (R4 for 3) → A_5 (R2 for 5) → A_6 (R1 for 3)
α	2	16	A_7 (R2 for 5) → A_8 (R2 for 3)
β	3	16	A_9 (R1 for 3) → A_{10} (R3 for 5)
β	4	8	A_{11} (R2 for 3) → A_{12} (R3 for 3)

Problem 17: 12563 solutions IVT = 0.88

Agent	Order	Deadline	Activities With Resource and Time Requirements
α	1	16	A_1 (R1 for 2) → A_2 (R4 for 3) → A_3 (R3 for 3)
α	2	20	A_4 (R4 for 4) → A_5 (R1 for 2) → A_6 (R4 for 5) → A_7 (R4 for 2) → A_8 (R3 for 5)
β	3	8	A_9 (R3 for 2) → A_{10} (R2 for 5)
β	4	16	A_{11} (R3 for 5) → A_{12} (R2 for 3)

Problem 18: 1042 solutions IVT = 0.48

Agent	Order	Deadline	Activities With Resource and Time Requirements
α	1	16	A_1 (R3 for 5) → A_2 (R1 for 2) → A_3 (R2 for 2) → A_4 (R2 for 3)
α	2	8	A_5 (R4 for 2) → A_6 (R1 for 2)
β	3	16	A_7 (R2 for 3) → A_8 (R4 for 5)
β	4	16	A_9 (R4 for 4) → A_{10} (R2 for 2) → A_{11} (R4 for 4) → A_{12} (R3 for 3)

6 Literature Cited

Corkill, D. D., and V. R. Lesser. 1983. The use of meta-level control for coordination in a distributed problem solving network. In *Eighth International Joint Conference on Artificial Intelligence*, pp. 748-756. Karlsruhe, FRG.

Dechter, R., and J. Pearl. 1988. Network-based heuristics for constraint-satisfaction problems. *Artificial Intelligence* **34**: 1-38.

Lander, S. E., and V. R. Lesser. 1992. Negotiated search: Organizing cooperative search among heterogeneous expert agents. In *Proceedings of the Fifth International Symposium on Artificial Intelligence, Applications in Manufacturing and Robotics*, pp. 351-358. Cancun, Mexico.

Lesser, V. R. 1991. A retrospective view of FA/C distributed problem solving. *IEEE Transactions on Systems, Man, and Cybernetics* **21**(6): 1347-1362.

Lesser, V. R., and L. D. Erman. 1980. Distributed interpretation: a model and an experiment. *IEEE Transactions on Computers* **C--29**(12): 1144-1163.

Minton, S., M. D. Johnston, A. B. Philips, and P. Laird. 1990. Solving large-scale constraint satisfaction and scheduling problems using a heuristic repair method. In *Proceedings of the Eighth National Conference on Artificial Intelligence*, pp. 17-24. Boston, MA: Morgan-Kaufmann.

Fox, M. S., N. Sadeh, and C. Baykan. 1989. Constrained heuristic search. In *Eleventh International Joint Conference on Artificial Intelligence*, pp. 309-315. Detroit, Michigan.

Purdom, P. 1983. Search rearrangement backtracking and polynomial average time. *Artificial Intelligence* **21**: 117-133.

Sokal, R. R., and F. J. Rohlf. 1981. *Biometry* (Second ed.). San Francisco: W.H. Freeman and Company.

Sycara, K., S. Roth, N. Sadeh, and M. Fox. 1991. Distributed constrained heuristic search. *IEEE Transactions on Systems, Man and Cybernetics* **21**(6): 1446-1461.

Yokoo, M., E. H. Durfee, T. Ishida, and K. Kuwabara. 1992. Distributed Constraint Satisfaction for Formalizing Distributed Problem Solving. In *Twelfth International Conference on Distributed Computing Systems*, pp. 614-621. Yokohama, Japan: IEEE.