Multi-HDCS: Solving DisCSPs With Complex Local Problems Cooperatively

David Lee, Inés Arana, Hatem Ahriz and Kit Hui
School of Computing, The Robert Gordon University
St. Andrew Street, Aberdeen AB25 1HG, UK
Email: {dl, ia, ha, khui}@comp.rgu.ac.uk

Abstract*— We propose Multi-HDCS, a new hybrid approach for solving Distributed CSPs with complex local problems. In Multi-HDCS, each agent concurrently: (i) runs a centralised systematic search for its complex local problem; (ii) participates in a distributed local search; (iii) contributes to a distributed systematic search. A centralised systematic search algorithm runs on each agent, finding all non-interchangeable solutions to the agent's complex local problem. In order to find a solution to the overall problem, two distributed algorithms which only consider the local solutions found by the centralised systematic searches are run: a local search algorithm identifies the parts of the problem which are most difficult to satisfy, and this information is used in order to find good dynamic variable orderings for a systematic search. We present two implementations of our approach which differ in the strategy used for local search: breakout and penalties on values. Results from an extensive empirical evaluation indicate that these two Multi-HDCS implementations are competitive against existing distributed local and systematic search techniques on both solvable and unsolvable distributed CSPs with complex local problems.

Keywords-**Distributed constraint satisfaction; local search; hybrid algorithms for distributed problem solving;**

I. Introduction

A Constraint Satisfaction Problem (CSP) consists of a set of variables, a corresponding set of domains (one per variable) and a set of constraints that restricts the values that variables may take concurrently. A CSP is solved when a value is assigned to each variable such that all constraints are satisfied. CSPs are normally solved by one of two main classes of algorithms: (i) Systematic search algorithms, which are complete and; (ii) Local search algorithms, which are incomplete but can be faster, for larger problems, than systematic algorithms [1].

A Distributed Constraint Satisfaction Problem (DisCSP) is a CSP which is divided into several inter-related local problems, each assigned to a different agent [2]. Thus, each agent has knowledge of the variables and corresponding domains of its local problem together with the constraints relating its own variables (intra-agent constraints) and the constraints linking its local problem to other local problems (inter-agent constraints). Agents, therefore, do not have a global view of the problem and must co-operate in order to solve the problem.

Traditionally, researchers have developed algorithms for DisCSPs where each local problem was fine-grained containing a single variable (i.e. each agent was responsible for only one variable) but new research has looked at DisCSPs where each agent is responsible for a complex local problem containing several variables [2]. A number of DisCSPs are naturally distributed containing a high number of intra-agent constraints with relatively few inter-agent constraints. For example, a University department can be primarily responsible for scheduling its own classes (the agent's complex local problem) but must check with other departments when a subject is taught to students belonging to other departments. It must also make other checks, for example, that the desired classrooms are free at the desired time.

In this work, we propose Multi-HDCS, a novel approach for the resolution of naturally distributed DisCSPs which considers the intra-agent constraints and inter-agent constraints separately using different search strategies concurrently. Multi-HDCS combines: (i) a centralised systematic algorithm for each complex local problem; (ii) a distributed local search algorithm and; (iii) a distributed systematic search algorithm. We present two implementations of our approach which differ in the strategy used during distributed local search: Multi-HDCS-Pen uses penalties on values [3] whilst Multi-HDCS-DB uses the breakout technique [4].

The remainder of this paper is structured as follows. Section II reviews related work. Section III presents the Multi-HDCS approach and our two implementations: Multi-HDCS-Pen and Multi-HDCS-DB. These implementations are evaluated against their main competitors in section IV. Finally, conclusions are drawn in section V.

II. Related Work

A variety of algorithms for fine-grained DisCSPs exist e.g. ABT, AWCS and DisBO [2]. These algorithms can be used to solve DisCSPs with complex local problems by creating virtual agents, one for each variable in the local problem an agent is responsible for. However, this prevents full use of the knowledge present in the agent. Consequently, both systematic and local search algorithms for the resolution of DisCSPs with complex local problems have been developed. Systematic search algorithms include Multi-AWCS [5] which uses a local AWCS solver to ensure the satisfaction of intra-agent constraints, whilst a global AWCS solver ensures the satisfaction of inter-agent constraints. Multi-ABT is a comparable extension for ABT [6].

Local search algorithms for DisCSPs with complex local problems differ in the strategy used to escape quasi-local-optima. DisBO-wd [7] attaches time-decaying weights to violated constraints whilst Multi-DisPeL [7] imposes penalties on values leading to a deadlock.

Recently, research into DisCSPs with complex local problems has focused on compilation formulation for existing distributed algorithms primarily for distributed constraint optimization [8]. This work focuses on generating solutions to an agent's local problem before solving the global problem.

Unlike local search algorithms, systematic algorithms are complete. However, local search algorithms tend to be faster for large problems. These two search types can be combined to gain the advantages of both approaches: completeness and fast(er) problem resolution. Multi-Hyb-Pen [9] is a two-phased hybrid approach for DisCSPs with complex local problems. In the first phase, it runs a centralised systematic search algorithm on each agent to find all non-interchangeable solutions to each complex local problem (i.e. those solutions of external relevance to an agent's local problem) whilst concurrently running a distributed local search algorithm to find a solution to the global problem. During the second phase, it runs a distributed systematic search algorithm if a solution has not been found during the first phase. Note that when the first phase lasts a long time, the distributed systematic search has to wait a long time before it can start finding solutions. This may be due, for example, to the time required to find all non-interchangeable solutions to a particularly under-constrained complex local problem with lots of non-interchangeable solutions. The Multi-HDCS approach presented below aims to combat this problem by allowing the systematic search to start at the same time as the local search. The main significant differences between the two approaches are highlighted after the presentation of Multi-HDCS.

III. Multi-HDCS

Multi-HDCS is a novel distributed hybrid approach for solving DisCSPs with complex local problems. Our approach runs a centralised systematic search and two distributed search algorithms concurrently to solve the problem. In order to explain the approach, a very simple timetabling problem is used, as illustrated in Figure 1. A university has three departments (Computing, Art and Business). Each department has a number of courses and a number of modules (subjects). A course hosted in one department can include a module taught by another department (external module). There are four potential time slots for modules to be scheduled: 9am, 10am, 11am and 12noon. Each department is responsible for issuing their own timetable. In order to produce appropriate timetables a department needs to check all its internal constraints (for example, modules belonging to the same course are taught at times which do not clash) and external constraints (for example, modules taught in courses external to the department and external modules taught in local courses have appropriate times).

In the **Multi-HDCS** approach (see Algorithm 1), each agent attempts to solve their own local problem using a centralised systematic search (step 5). This search finds all solutions to an agent's complex local problem which are externally relevant, i.e. all non-interchangeable solutions which satisfy all intra-agent constraints and are distinguishable when looking only at externally relevant variables (involved in inter-agent constraints). In the timetabling example above, a centralised systematic search would be run for each department. If any department is unable to find a solution to its complex local problem, the problem is unsolvable. Otherwise, after all agents have found at least one solution to their local problem two additional distributed searches (local search and systematic search) are started concurrently (steps 7 and 8), which attempt to find a solution which satisfies all inter-agent constraints. The centralised systematic search (step 5) continues to run and dynamically communicates new value combinations to both distributed searches. Meanwhile, the distributed local search (step 7) identifies difficult parts of the problem and passes this information to the distributed systematic search (step 8) at synchronisation points, to be used for dynamic variable ordering.

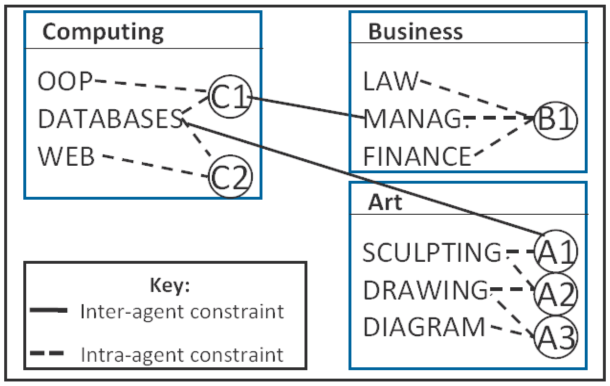

Fig. 1. Sample Scheduling Problem

Algorithm 1 Multi-HDCS

```
1:  Initialise each agent with its complex local problem
2:  while not($solved$) concurrently do
3:      for each agent $a_i$ with local problem $lp_i$ concurrently do
4:          Run a centralised systematic search to find all externally
            relevant solutions to $lp_i$
5:      end for
6:      Run a distributed local search combining local problem
        solutions (found in step 5), checking inter-agent constraints
        only. Regularly pass the knowledge learnt during search to
        the distributed systematic search below (step 8)
7:      Run a distributed systematic search combining local prob-
        lem solutions (found in step 5) and using distributed local
        search's findings (from step 7) to dynamically order agents
8:  end while
9:  if distributed local search or distributed systematic search has
    found solution $S$ then
10:     return $S$
11: else
12:     Return "Unsolvable problem" as either the centralised system-
        atic search (step 5) or the distributed systematic search (step
        8) has detected unsolvability
13: end if
```

Completeness: the centralised systematic searches (step 5) are guaranteed to find all non-interchangeable solutions to the complex local problems. If one of these problems does not have a solution, Multi-HDCS will detect unsolvability. If, however, all complex local problems have at least one solution the distributed systematic search (step 8) will either find a solution or detect unsolvability. Note that the distributed systematic search can complete its run whilst the centralised systematic search is still running if a solution to the global problem is found. However, for unsolvable problems it does not terminate until all centralised systematic searches have found all non-interchangeable solutions or one has detected unsolvability.

Termination: each instance of the centralised systematic search terminates when either: (i) it has found all non-interchangeable solutions to its local problem; (ii) it detects the unsolvability of its local problem and has informed all other agents; (iii) receives a message from one of the agents stating that the problem is unsolvable; (iv) receives a message from either the distributed local search or the distributed systematic search stating that the problem has been solved. The distributed local search stops when either: (i) it has found a solution or; (ii) the distributed systematic search has either found a solution or detected unsolvability. The distributed systematic search terminates when either: (i) it has found a solution;(ii) it has detected that the problem is unsolvable once the centralised systematic searches have completed their search; (iii) the distributed local search has found a solution. Since the centralised systematic searches, the distributed local search and the distributed systematic search terminate, Multi-HDCS also terminates.

We present two implementations of our approach: Multi-HDCS-Pen and Multi-HDCS-DB.

A. Multi-HDCS-Pen

Multi-HDCS-Pen runs SEBJ [9] as the centralised systematic search algorithm, InterDisPeL (see below) as the distributed local search algorithm and InterPODS (see below) as the distributed systematic search algorithm.

InterDisPeL is a penalty-based distributed local search algorithm inspired in Multi-DisPeL [7]. Unlike Multi-DisPeL, InterDisPeL: (i) considers only inter-agent constraints; (ii) only chooses variable-value combinations approved by SEBJ; (iii) maintains, for each agent, an overall count of the penalties it has imposed in the spirit of [10]. Thus, whenever a penalty is imposed on an agent's variable, the agent's penalty count is increased. This allows InterDisPeL to detect the complex local problems that are difficult to solve (i.e. with high penalties) and inform InterPODS (see below).

InterPODS (see Algorithms 2 and 3) is a new systematic algorithm for solving inter-agent constraints which uses complex variables. InterPODS is inspired by the much simpler PenDHyb algorithm [10] with substantial differences: (i) each InterPODS agent knows only those value combinations which are compatible with the local problem's intra-agent constraints; (ii) InterPODS only considers inter-agent constraints; (iii) InterPODS uses complex variables; (iv) the next agent for processing is chosen dynamically based on the maximum degree heuristic, the minimum domain heuristic and each agent's penalty count obtained from the concurrent InterDisPeL search. For example, assuming that maximum degree and minimum domain were the same for the agents representing $computing$ and $business$ then if agent art has already selected a value for its complex variable and $computing$ and $business$ have penalties of 0 and 3, InterPODS will select the $business$ agent for processing. The penalty information is synchronized with InterDisPeL's current penalty counts regularly[1].

Algorithm 2 InterPODS

```
1: initialise agents with partial solutions from centralised systematic
   search as its domain
2: set $first\_agent$ and $curr\_agent$ to highest agent in ordering
   schema.
3: ChooseVal($curr\_agent$)
4: while messages exist do
5:    if receive backjumping message with $backjumping\_agent$
      then
6:       ChooseVal ($backjumping\_agent$)
7:    else if receive cpa message with $next\_agent$ then
8:       ChooseVal ($next\_agent$)
9:    end if
10: end while
```

B. Multi-HDCS-DB

Multi-HDCS-DB runs SEBJ [9] as the centralised systematic search algorithm, InterDisBO-wd (see below) as the distributed local search algorithm and InterPODS as the distributed systematic search algorithm.

InterDisBO-wd is inspired by the breakout-based algorithm DisBO-wd [7]. Unlike DisBO-wd, InterDisBO-wd: (i) checks only inter-agent constraints; (ii) considers only variable-value combinations approved by SEBJ; (iii) maintains, for each agent, a cumulative constraint-weight counter, i.e. the sum of the weights on all constraints which involve one of the agent's variables. These counters enable the identification of complex local problems which are difficult to solve (i.e. with high constraint weights) to guide the InterPODS systematic search.

InterPODS has already been presented above for Multi-HDCS-Pen. The version of InterPODS used in Multi-HDCS differs only in that the next agent for processing is now chosen dynamically based on each agent's constraint weight from the concurrent InterDisBO-wd search tiebroken with minimum domain and maximum degree heuristics. The constraint weight information is synchronized with InterDisBO-wd's current constraint weights regularly[2].

[1]Extensive empirical results suggest a synchronisation every 2 cycles.
[2]Extensive empirical results suggest a synchronisation every 30 cycles.

Algorithm 3 procedure ChooseVal($curr\_agent$)

```
1: for each value $d_i$ in agent $curr\_agent$'s domain do
2:     if all higher priority constraints are satisfied then
3:         if all higher priority nogoods are not consistent with agent values then
4:             assign value $d_i$ representing the chosen local solution for that agent's problem to agent $curr\_agent$ in $cpa$
5:             set $next\_agent$ to next agent dynamically chosen from ordering schema.
6:             if $next\_agent = last\_agent$ then
7:                 return "solution found"
8:             end if
9:             send message to $next\_agent$ with $cpa$
10:        end if
11:    else if higher priority constraints are violated then
12:        for each higher priority constraint which is violated do
13:            record the agent and value pair as part of a nogood value $d_i$ to agent $curr\_agent$
14:        end for
15:    end if
16: end for
17: if centralised systematic search has found new solutions then
18:     synchronize domain with centralised systematic search.
19:     remove nogoods containing agents who have new values.
20:     restart search from this agent.
21: end if
22: if local search has updated penalty counts then
23:     synchronize information from local search.
24: end if
25: if $curr\_agent$ is $first\_agent$ and has no assigned value and centralised systematic search has terminated then
26:     return "unsolvable problem"
27: else if $curr\_agent$ has no assigned value then
28:     Create a conflict set for agent $curr\_agent$ containing all agents involved in nogoods for values belonging to agent $curr\_agent$
29:     Send a backjump message to the lowest priority agent in the conflict set.
30: end if
```

C. *Multi-HDCS vs. Multi-Hyb*

Both Multi-HDCS and Multi-Hyb-Pen use one centralised systematic search per agent, one distributed local search and one distributed systematic search. However, their overall approaches are substantially different as follows: (i) In Multi-HDCS all three types of searches run concurrently whereas in Multi-Hyb-Pen a two-phase strategy is used; (ii) In Multi-HDCS, the knowledge discovered during the distributed local search is regularly passed to the distributed systematic search; (iii) Multi-Hyb-Pen uses a fixed-order distributed systematic search whereas Multi-HDCS dynamically orders its agents in its distributed systematic search; (iv) in Multi-HDCS variable domains are dynamic for the distributed systematic search whereas in Multi-Hyb, these are static.

IV. Experimental Evaluation

An extensive experimental evaluation of Multi-HDCS-Pen and Multi-HDCS-DB on both solvable and unsolvable problems has been carried out. For solvable problems they were compared against the following leading algorithms for DisCSPs with complex local problems: Multi-ABT [6], Multi-AWCS [5], Multi-Hyb-Pen [9], Multi-DisPeL and DisBO-wd [7]. For unsolvable problems, Multi-HDCS-Pen and Multi-HDCS-DB were compared against Multi-ABT, Multi-AWCS and Multi-Hyb-Pen. Note that a comparison with Multi-DisPeL and DisBO-wd for unsolvable problems is not appropriate since these two algorithms are incomplete.

We verified the implementations of our comparison algorithms as follows: Multi-ABT with the distributed graph colouring problems in [6], Multi-AWCS with the distributed randomly generated experiments reported in [5], Multi-Hyb-Pen with the distributed randomly generated experiments reported in [9] and Multi-DisPeL and DisBO-wd with the distributed randomly generated problems reported in [7]. In all cases, the results were at least as good as those reported by their authors.

100 different instances for each problem type (with the same ratio of intra-agent to inter-agent constraints) were solved and the established metrics in the field were measured: (i) average and median number of messages sent between agents and; (ii) average and median number of non-concurrent constraint checks (NCCCs) performed. Note that the number of messages/NCCCs required for termination detection are not included in the results for any of the algorithms as this is common practice in the field, e.g. [5]. Whilst CPU time is not an established measure for comparing DisCSP algorithms [11], our CPU time results matched the trends of other measures.

For Multi-DisPeL and DisBO-wd the percentage of problems solved was measured, and they solved most of the problems. For those experiments where the algorithms were unable to find a solution to one or more problems, this is indicated by a * in the results tables (see below) and the effort wasted is not included in the results. A cutoff of 100n iterations (where n is the number of variables) for Multi-DisPeL and 200n iterations for DisBO-wd (since 2 DisBO-wd cycles of *improve* and *ok*? are equal to 1 Multi-DisPeL cycle) was used.

In our experiments, we generally used naturally distributed problems i.e. problems which have a high ratio of intra-agent to inter-agent constraints from 90:10 to 65:35. The number of variables used ranged from 25 to 200. We used a variety of problem types, including: (i) distributed randomly generated problems; (ii) distributed 3-colour graph colouring problems; (iii) distributed scheduling problems and; (iv) distributed sensor network problems. Note that the sensor network problems are not naturally distributed since they have a relative simple local problem within each agent and many inter-agent constraints (see below). However, we include these in our comparison to determine the limitations of our approach.

Solvable Problems

Randomly Generated Problems: Table I presents the median values for solvable randomly generated problems using 5 agents, a domain size of 8, a constraint density of 0.2 and constraint tightness of 0.35. For medium-sized problems (60 to 125 variables), Multi-HDCS-DB performed best for number of messages. For problems with 125 or more variables, Multi-Hyb-Pen had the smallest number of messages, although the number of messages was also very small for Multi-HDCS-DB and Multi-DisPeL. For NCCCs, Multi-HDCS-DB again gave the best results for most problems.

Graph Colouring Problems: Median results for 3-colour distributed graph colouring problems with 150 to 200 nodes, 15 to 25 agents and a degree between 4.9 and 5.1 are presented in Table II. Multi-Hyb-Pen gives best results for the number of messages with Multi-HDCS-DB in 2nd place. For most instances Multi-HDCS-DB is the best performing algorithm for NCCCs.

Scheduling Problems: The scheduling problems used are based on the generator in [12]. Specifically, each department (agent) needs to hold several meetings (variables). Each meeting is attended by a number of people. Each department has at least one location where meetings can be held and employees from another department can attend meetings in this location provided they have enough travelling time. There are three types of constraints: (i) inequality constraints between all meetings held in the same department; (ii) travelling time constraints between inter-departmental meetings with one or more common participants and; (iii) precedence constraints between meetings. We experimented with solvable scheduling problems with 50-80 meetings, 5 departments (agents), timeframes of 6 or 7 units and a constraint density of 0.18. The percentage of intra-agent constraints varied between 85% and 90%. The distance between two departments with common meetings was of between 1 and 3 time units.

Table III shows the median results obtained for solvable scheduling problems. Multi-Hyb-Pen performs best for number of messages followed by Multi-HDCS-DB and Multi-DisPeL. For NCCCs, Multi-ABT performed best followed by Multi-Hyb-Pen. It should be noted that in a number of instances SEBJ very quickly found all local solutions meaning that local search had very little time to gather initial ordering information for InterPODS.

Sensor Networks: Finally, our algorithms were evaluated on Grid-based Sensor Network DisCSPs [13]. These problems are not naturally distributed since they have a large number of inter-agent constraints combined with relatively simple local problems for each agent, i.e. they contain 15% intra-agent constraints and 85% inter-agent constraints. They provide an interesting case to determine whether the Multi-HDCS approach also functions for problems which are not naturally distributed. The problems used had 5 targets, between 25 and 64 sensors (grids of 5, 6, 7, 8), k-visibility of 2, k-compatibility of 1, probability of visibility of 0.9 and probability of compatibility of 0.6.

Median results for Multi-HDCS-Pen and Multi-HDCS-DB are shown in Table IV. For the number of messages, Multi-HDCS-DB, Multi-Hyb-Pen and Multi-ABT all perform best for different problem combinations although Multi-HDCS-DB offers the most consistent performance. Multi-HDCS-Pen performed best for NCCCs with the exception of one instance where Multi-AWCS was better.

TABLE I
MEDIAN RESULTS ON SOLVABLE RANDOMLY GENERATED PROBLEMS FOR MULTI-HDCS-PEN (M-HDCS-PEN) AND MULTI-HDCS-DB (M-HDCS-DB) AGAINST MULTI-HYB-PEN (M-HYB-PEN), MULTI-ABT (M-ABT), MULTI-AWCS (M-AWCS), MULTI-DISPEL (M-DISPEL), AND MULTI-DISBO-WD (M-BO-WD).

		Median number of messages						
n	**% intra :inter con.**	**m-hdcs -pen**	**m-hdcs -db**	**m-hyb -pen**	**m-abt**	**m-awcs**	**m-dispel**	**m-bo-wd**
60	90:10	234	**60**	399	842	4834	536	1150*
60	80:20	344	**85**	197	1692	5287	422	1165
60	70:30	278	**156**	818	6832	4475	496	985
70	80:20	130	**45**	159	731	3672	208	435
70	70:30	264	**60**	112	1141	3907	194	420
80	80:20	70	**42**	143	440	3991	104	335
80	70:30	117	**38**	89	500	6076	108	295
90	80:20	70	**35**	94	336	4242	66	275
90	70:30	125	**35**	81	298	6193	80	265
100	80:20	70	**35**	56	248	5922	56	235
100	70:30	70	**35**	78	276	7235	60	225
125	80:20	70	35	**20**	197	6297	40	225
125	70:30	70	**35**	60	152	9218	40	205
150	80:20	70	35	**20**	152	6803	28	215
150	70:30	70	35	**30**	128	14554	32	195
175	80:20	70	35	**20**	134	10707	24	210
175	70:30	70	35	**20**	118	15126	24	190
		Median number of NCCCs						
n	**% intra :inter con.**	**m-hdcs -pen**	**m-hdcs -db**	**m-hyb -pen**	**m-abt**	**m-awcs**	**m-dispel**	**m-bo-wd**
60	90:10	**59560**	60088	163585	314067	165118	1187335	469162*
60	80:20	75413	**71387**	277408	420384	277408	949616	440862
60	70:30	1012213	537988	2761171	286821	**182936**	1148704	353862
70	80:20	50698	**49960**	151678	284713	124238	745608	252678
70	70:30	88373	**85467**	291421	524487	135090	673099	244962
80	80:20	**48123**	49126	118874	207389	149599	588111	283827
80	70:30	56643	**56339**	169884	356405	265274	606084	262707
90	80:20	46855	**45307**	117668	278057	177570	611811	308444
90	70:30	52380	**51510**	140181	224968	291656	638729	299228
100	80:20	44687	**44571**	107836	214806	285431	690977	339423
100	70:30	**50638**	52368	132031	265460	385969	690455	324668
125	80:20	46992	**46706**	106435	185646	357508	952787	509090
125	70:30	51280	**50360**	125553	360376	600688	936775	485739
150	80:20	45587	**45250**	100020	235880	441287	1362161	728427
150	70:30	54756	**52613**	120105	268777	1302570	1281866	682116
175	80:20	45774	**45613**	98875	155900	885339	1926771	976712
175	70:30	51805	**50468**	110325	235168	1453996	1831216	908710

A. Unsolvable Problems

Our experiments with unsolvable problems distinguish between two categories of unsolvable problems: (i) those where at least one complex local problem is unsolvable and; (ii) those where all complex local problems are solvable, but no overall solution exists.

Randomly Generated Problems: Median results for unsolvable randomly generated problems using 5 agents, a domain size of 8 and a constraint tightness of 0.35 are presented for problems which have one or more complex local problems that are unsolvable. For these problems, the number of messages for Multi-HDCS-Pen, Multi-HDCS-DB, Multi-Hyb-Pen and Multi-ABT is very low, since unsolvability is detected locally. Therefore, only results for NCCCs are presented in Table V. For constraint checks, Multi-HDCS-Pen,

TABLE II
MEDIAN RESULTS ON SOLVABLE GRAPH COLOURING PROBLEMS MULTI-HDCS-PEN (M-HDCS-PEN) AND MULTI-HDCS-DB (M-HDCS-DB) AGAINST MULTI-HYB-PEN (M-HYB-PEN), MULTI-ABT (M-ABT), MULTI-AWCS (M-AWCS), MULTI-DISPEL (M-DISPEL), AND MULTI-DISBO-WD (M-BO-WD).

				Median number of messages						
n node	n ag.	**deg**	**intra: inter con.**	**m-hdcs -pen**	**m-hdcs -db**	**m-hyb -pen**	**m-abt**	**m-awcs**	**m-dispel**	**m-bo-wd**
150	15	4.9	90:10	486	120	**40**	490	1281	595	855
150	15	5.1	90:10	481	120	**35**	608	1437	714	840*
150	15	4.9	85:15	482	135	**88**	1324	2011	791	3411*
150	15	5.1	85:15	498	134	**99**	1539	2068	749	1020*
150	25	4.9	90:10	1205	200	**35**	373	1508	1176	1175*
150	25	5.1	90:10	1182	200	**29**	399	1534	1176	1200*
150	25	4.9	85:15	1858	200	**44**	1133	2161	1548	1425*
150	25	5.1	85:15	1257	200	**41**	1104	2360	1380	975*
200	20	4.9	90:10	842	160	**62**	698	2146	1197	1420*
200	20	5.1	90:10	832	160	**73**	938	2328	1216	1300*
200	20	4.9	85:15	822	178	**138**	2304	3262	1482	1420*
200	20	5.1	85:15	805	**169**	181	3299	3310	1634	1600*
200	25	4.9	90:10	1253	200	**51**	657	2350	1716	1425*
200	25	5.1	90:10	1253	200	**45**	869	2092	1800	1575*
200	25	4.9	85:15	1301	203	**85**	2541	3202	1908*	1900*
200	25	5.1	85:15	1277	204	**98**	2776	3306	1896	1975
				Median number of NCCCs						
n node	n ag.	**deg**	**intra: inter con.**	**m-hdcs -pen**	**m-hdcs -db**	**m-hyb -pen**	**m-abt**	**m-awcs**	**m-dispel**	**m-bo-wd**
150	15	4.9	90:10	1387	**1185**	3579	1266	3172	46215	66583
150	15	5.1	90:10	1449	**1255**	3689	1589	3435	57967	63567*
150	15	4.9	85:15	2392	2402	7011	**1972**	3938	56778	78811*
150	15	5.1	85:15	2570	**2375**	6796	2478	3878	59313	73285*
150	25	4.9	90:10	570	**423**	675	689	1454	30961	53242*
150	25	5.1	90:10	541	**459**	633	724	1417	33134	53127*
150	25	4.9	85:15	1163	**791**	944	1132	1732	40156	52768*
150	25	5.1	85:15	1017	**837**	887	1042	1795	37087	41801*
200	20	4.9	90:10	1605	1461	4195	**1434**	3836	71275	104597*
200	20	5.1	90:10	1530	**1438**	4403	1716	4185	70314	105865*
200	20	4.9	85:15	2609	2775	8277	**2486**	4454	81720	96362*
200	20	5.1	85:15	**2498**	2539	8184	3171	4522	90557	108954*
200	25	4.9	90:10	997	**751**	1843	1014	2723	61481	87216*
200	25	5.1	90:10	998	**769**	1703	1214	2499	68940	99001*
200	25	4.9	85:15	1920	**1399**	2986	1969	3067	67226*	100466*
200	25	5.1	85:15	1766	**1471**	2920	2044	3154	68374	108364

Multi-HDCS-DB and Multi-Hyb-Pen perform best. Note that they all give identical results because of their common SEBJ searches.

We also conducted experiments for problems that had solutions to all complex local problems but no global solution with identical parameters the results of which are also shown in Table V. Multi-HDCS-DB significantly outperforms all other algorithms both for number of messages and for NCCCs.

Graph Colouring Problems: Median results for unsolvable 3-colour distributed graph colouring problems with 150 to 200 nodes, 15 to 25 agents and 4.9 to 5.1 degree are presented in table VII. For graph colouring problems, most complex local problems had solutions but not all. Therefore, whilst the hybrid algorithms are able to detect unsolvability with very few messages (only those required for termination detection), Multi-ABT now incurs a lot of messages trying to find consistent values for those agents which have problems that have

TABLE III
MEDIAN RESULTS ON SOLVABLE SCHEDULING PROBLEMS FOR MULTI-HDCS-PEN (M-HDCS-PEN) AND MULTI-HDCS-DB (M-HDCS-DB) AGAINST MULTI-HYB-PEN (M-HYB-PEN), MULTI-ABT (M-ABT), MULTI-AWCS (M-AWCS), MULTI-DISPEL (M-DISPEL), AND MULTI-DISBO-WD (M-BO-WD).

			Median number of messages						
n meet	**n time**	**intra: inter con.**	**m- -hdcs -pen**	**m- -hdcs -db**	**m- -hyb -pen**	**m- -abt**	**m- -awcs**	**m- dispel**	**m- bo-wd**
50	7	90:10	65	50	**20**	81	340	68	295*
50	7	85:15	70	50	**40**	112	381	60*	405*
50	6	90:10	65	50	**10**	64	269	52	155*
50	6	85:15	65	45	**20**	85	278	56	185*
60	7	90:10	65	50	**20**	86	359	64	245*
60	7	85:15	70	50	**20**	116	396	64	285*
60	6	90:10	65	50	**10**	78	288	32	145*
60	6	85:15	65	40	**20**	92	289	40	185*
70	7	90:10	68	50	**20**	103	380	44	235*
70	7	85:15	70	45	**20**	121	413	52	275*
70	6	90:10	65	40	**20**	91	274	40	165*
70	6	85:15	65	40	**20**	108	290	40	215*
80	7	90:10	70	50	**20**	115	404	48	235
80	7	85:15	70	45	**20**	136	403	52	285*
80	6	90:10	65	40	**20**	98	284	32	185
80	6	85:15	65	40	**20**	108	335	36	220
			Median number of NCCCs						
n meet	**n time**	**intra: inter con.**	**m- -hdcs -pen**	**m- -hdcs -db**	**m- -hyb -pen**	**m- -abt**	**m- -awcs**	**m- dispel**	**m- bo-wd**
50	7	90:10	7571	7571	7162	**6988**	7309	112308	110290*
50	7	85:15	12336	12086	8807	**8584**	7883	98814*	97449*
50	6	90:10	3592	3592	**2933**	3793	5534	73805	57262*
50	6	85:15	4399	4399	**4266**	4456	5424	73241	63277*
60	7	90:10	12833	12833	10777	10901	**10613**	160589	158103*
60	7	85:15	17294	17253	12945	**10795**	**10795**	153688	150166*
60	6	90:10	5948	5948	**5095**	5490	7894	89497	91349*
60	6	85:15	5817	5817	**5725**	5783	7856	93741	96019*
70	7	90:10	18496	18255	15377	**13044**	13739	198303	203387*
70	7	85:15	19745	20453	15571	**12857**	14865	204447	191004*
70	6	90:10	7585	7585	**6586**	6906	10373	131723	136478*
70	6	85:15	7891	7891	7602	**6499**	18715	132870	140313*
80	7	90:10	20834	20432	17434	**14685**	18715	270668	280138
80	7	85:15	21770	22551	18086	**15358**	18321	266825	262763*
80	6	90:10	8587	8407	8863	**7432**	14264	177645	187330
80	6	85:15	8327	8282	8268	**7044**	14635	176676	193495

TABLE IV
MEDIAN RESULTS FOR SOLVABLE GRID-BASED SENSOR NETWORK PROBLEMS FOR MULTI-HDCS-PEN (M-HDCS-PEN) AND MULTI-HDCS-DB (M-HDCS-DB) AGAINST MULTI-HYB-PEN (M-HYB-PEN), MULTI-ABT (M-ABT), MULTI-AWCS (M-AWCS), MULTI-DISPEL (M-DISPEL), AND MULTI-DISBO-WD (M-BO-WD).

	Median number of messages						
num sen-sors	**m-hdcs -pen**	**m-hdcs -db**	**m-hyb -pen**	**m- -abt**	**m-awcs**	**m-dispel**	**m-bo-wd**
25	145	**50**	69	204	299	80*	575*
36	145	**40**	50	52	185	40*	285*
49	85	40	25	**24**	94	40*	160*
64	85	40	**14**	19	101	28*	120*
	Median number of NCCCs						
num sen-sors	**m-hdcs -pen**	**m-hdcs -db**	**m-hyb -pen**	**m- -abt**	**m-awcs**	**m-dispel**	**m-bo-wd**
25	**2727**	4716	4072	8859	5959	40031*	66968*
36	**2337**	2512	2936	4329	3888	25707*	31359*
49	**2254**	2374	2708	2755	2314	18280*	19366*
64	2371	2373	2541	2294	**1856**	14432*	15397*

solutions. Consequently only results for NCCCs are presented in Table VI.

Since Multi-HDCS-Pen and Multi-HDCS-DB only execute the centralised systematic search to detect unsolvability in this case, it is optimal for messages. The optimal algorithm for NCCCs varies according to the problem parameters either the hybrid algorithms or Multi-ABT.

TABLE V
MEDIAN RESULTS FOR UNSOLVABLE RANDOM PROBLEMS WITH ONE OR MORE AGENTS HAVING NO SOLUTION TO THEIR LOCAL PROBLEM. ALGORITHMS USED ARE MULTI-HDCS-PEN (M-HDCS-PEN), MULTI-HDCS-DB (M-HDCS-DB) MULTI-HYB-PEN (M-HYB-PEN), MULTI-ABT (M-ABT) AND MULTI-AWCS (M-AWCS).

Problems where at least one complex local problem has no solution							
			Median number of NCCCs				
n	**dens.**	**% intra :inter**	**hdcs pen**	**hdcs db**	**hyb pen**	**hyb abt**	**m- awcs**
60	0.2	90:10	**35438**	**35438**	**35438**	78010	8460541
70	0.2	80:20	**33927**	**33927**	**33927**	85986	9126579
70	0.2	70:30	**36562**	**36562**	**36562**	117992	11127918
80	0.2	80:20	**33916**	**33916**	**33916**	86237	11362216
80	0.2	70:30	**35072**	**35072**	**35072**	108608	14331052
90	0.2	80:20	**33239**	**33239**	**33239**	84951	14592206
90	0.2	70:30	**34284**	**34284**	**34284**	107228	17912129
100	0.2	80:20	**32649**	**32649**	**32649**	89197	18093057
100	0.2	70:30	**35310**	**35310**	**35310**	114103	22217753
125	0.2	80:20	**32556**	**32556**	**32556**	84950	29202074
125	0.2	70:30	**34538**	**34538**	**34538**	109136	35119651
150	0.2	80:20	**33056**	**33056**	**33056**	91300	42442786
150	0.2	70:30	**36430**	**36430**	**36430**	110922	50882090
175	0.2	80:20	**36156**	**36156**	**36156**	87780	58091327
175	0.2	70:30	**33210**	**33210**	**33210**	104237	68798883

TABLE VI
MEDIAN RESULTS FOR UNSOLVABLE RANDOM PROBLEMS WITH ALL AGENTS HAVING AT LEAST ONE SOLUTION TO THEIR LOCAL PROBLEM. ALGORITHMS USED ARE MULTI-HDCS-PEN (M-HDCS-PEN), MULTI-HDCS-DB (M-HDCS-DB) MULTI-HYB-PEN (M-HYB-PEN), MULTI-ABT (M-ABT) AND MULTI-AWCS (M-AWCS).

Problems where all complex local problems have a solution							
			Median number of messages				
n	**dens.**	**% intra :inter**	**hdcs pen**	**hdcs db**	**hyb pen**	**hyb abt**	**m- awcs**
60	0.2	80:20	703	**69**	177	762	33930
60	0.2	70:30	480	**69**	249	3950	41712
70	0.18	70:30	418	**54**	114	1266	48433
80	0.16	70:30	823	**49**	106	1242	55324
90	0.14	70:30	500	**56**	158	1968	61541
100	0.13	70:30	674	**49**	129	840	68524
			Median number of NCCCs				
n	**dens.**	**% intra :inter**	**hdcs pen**	**hdcs db**	**hyb pen**	**hyb abt**	**m- awcs**
60	0.2	80:20	**53186**	**53186**	62205	127460	7620027
60	0.2	70:30	113114	**83564**	251012	226011	7996729
70	0.18	70:30	102594	**91343**	136748	192851	10569556
80	0.16	70:30	135582	**124409**	174461	230568	13527324
90	0.14	70:30	254904	**238437**	374569	333709	16092489
100	0.13	70:30	330553	**298966**	362227	347370	19929678

Scheduling Problems: Unsolvable scheduling problems with 50-80 meetings, 5 departments (agents), a timeframe of 6 or 7 time units and a constraint density of 0.18 were conducted. The percentage of intra-agent constraints varied between 85% and 90%. Two departments with common meetings have a distance of between 1 and 3 time units. The results (not shown here) show that scheduling problems generally had no solution to the complex local problems and therefore all algorithms except Multi-AWCS incurred very low message costs and performed similarly on constraint checks.

TABLE VII
MEDIAN RESULTS FOR UNSOLVABLE GRAPH COLOURING PROBLEMS WITH ONE OR MORE AGENTS HAVING NO SOLUTION TO THEIR LOCAL PROBLEM. ALGORITHMS USED ARE MULTI-HDCS-PEN (M-HDCS-PEN), MULTI-HDCS-DB (M-HDCS-DB) MULTI-HYB-PEN (M-HYB-PEN), MULTI-ABT (M-ABT) AND MULTI-AWCS (M-AWCS).

				Median number of messages				
n node	**n ag.**	**deg**	**intra: inter con.**	**m- hdcs -pen**	**m- hdcs -db**	**m- hyb -pen**	**m- abt**	**m- awcs**
150	15	4.9	80:20	**0**	**0**	**0**	860	6307
150	15	5.1	80:20	**0**	**0**	**0**	947	6456
150	15	4.9	70:30	**0**	**0**	**0**	2911	9356
150	15	5.1	70:30	**0**	**0**	**0**	1899	9474
150	25	4.9	70:30	**0**	**0**	**0**	1576	13728
150	25	5.1	70:30	**0**	**0**	**0**	1630	14031
200	20	4.9	80:20	**0**	**0**	**0**	1277	9163
200	20	5.1	80:20	**0**	**0**	**0**	1497	9195
200	20	4.9	70:30	**0**	**0**	**0**	2296	14107
200	20	5.1	70:30	**0**	**0**	**0**	1956	14680
				Median number of NCCCs				
n node	**n ag.**	**deg**	**intra: inter con.**	**m- hdcs -pen**	**m- hdcs -db**	**m- hyb -pen**	**m- abt**	**m- awcs**
150	15	4.9	80:20	1525	1525	1525	**1202**	11590
150	15	5.1	80:20	1421	1421	1421	**1286**	11924
150	15	4.9	70:30	**2332**	**2332**	**2332**	2395	15259
150	15	5.1	70:30	2114	2114	2114	**1797**	15255
150	25	4.9	70:30	**296**	**296**	**296**	767	11580
150	25	5.1	70:30	**294**	**294**	**294**	758	11725
200	20	4.9	80:20	1415	1415	1415	**1304**	11910
200	20	5.1	80:20	1717	1717	1717	**1321**	11812
200	20	4.9	70:30	2512	2512	2512	**1727**	15300
200	20	5.1	70:30	2253	2253	2253	**1656**	15854

Sensor Network Problems: Table VIII shows median results for unsolvable sensor networks problems with 5 targets, 25-64 sensors (grids of 5, 6, 7 and 8), k-visibility of 2, k-compatibility of 1, probability of visibility of 0.9 and probability of compatibility of 0.3. The ratio of intra-agent to inter-agent constraints is 15% to 85%.

V. CONCLUSIONS

Multi-HDCS is a new hybrid approach for solving DisCSPs with complex local problems where the problem solving is carried out by concurrent cooperative searches: (i) a set of centralised systematic searches (one per agent) finds all non-interchangeable solutions to each agent's local problem; (ii) a distributed local search attempts to solve the inter-agent constraints using variable-value combinations approved by the centralised systematic searches. It also identifies local problems which are difficult to solve and passes this information to a distributed systematic search (see below); (iii) a distributed systematic search attempts to find a solution satisfying the inter-agent constraints using only variable-value combinations approved by centralised systematic searches whilst dynamically prioritising agents according to the level of difficulty of

TABLE VIII
MEDIAN RESULTS ON UNSOLVABLE GRID-BASED SENSOR NETWORK PROBLEMS. ALGORITHMS USED ARE MULTI-HDCS-PEN (M-HDCS-PEN), MULTI-HDCS-DB (M-HDCS-DB) MULTI-HYB-PEN (M-HYB-PEN), MULTI-ABT (M-ABT) AND MULTI-AWCS (M-AWCS).

	Median number of messages				
num sen-sors	**m-hdcs -pen**	**m-hdcs -db**	**m-hyb -pen**	**m- -abt**	**m-awcs**
25	1733	**262**	1293	2309	5524
36	2505	**331**	875	864	4657
49	2349	**300**	1006	680	3346
64	2052	**265**	554	381	3043
	Median number of NCCCs				
num sen-sors	**m-hdcs -pen**	**m-hdcs -db**	**m-hyb -pen**	**m- -abt**	**m-awcs**
25	**13536**	21870	22275	49663	92273
36	**11340**	12587	15229	24703	77773
49	10917	**8393**	22827	22292	64488
64	10170	**4887**	9225	13227	61675

their local problems assigned by the distributed local search. While Multi-HDCS may appear to be similar to Multi-Hyb, it differs from it substantially (see table IX).

TABLE IX
DIFFERENCES BETWEEN MULTI-HYB AND MULTI-HDCS.

	Multi-Hyb	**Multi-HDCS**
Phases	2	1
Concurrency	Centralised systematic search and distributed local search	All three types of search
Variables	Distributed local search uses complex variables	Distributed local search uses all externally relevant (single) variables
Domains	Static for Distributed systematic search.	Dynamic for Distributed systematic search.
Agent ordering	Static for distributed systematic search	Dynamic for distributed systematic search.
Communication between distributed local search and distributed systematic search	Once	Regularly

We have presented two implementations of Multi-HDCS: Multi-HDCS-Pen and Multi-HDCS-DB. These differ mainly in the algorithm used for distributed local search: Multi-HDCS-Pen uses a penalty-based algorithm (InterDisPeL) whereas Multi-HDCS-DB uses a breakout-based (i.e. weights on constraints) algorithm (Inter-DisBO-wd). Both algorithms use SEBJ to solve the agent's local problem and InterPODS as the distributed systematic search algorithms.

Substantial empirical results on several problem classes demonstrate that the Multi-HDCS approach (particularly in the Multi-HDCS-DB implementation) is generally competitive when compared to leading DisCSPs with complex local problems algorithms on both solvable problems and unsolvable problems.

Further implementations of the Multi-HDCS framework are possible. The algorithm for centralised systematic searches could be replaced by any complete centralised algorithm which finds all non-interchangeable solutions to a complex local problem. Moreover, different agents could use different algorithms to find their non-interchangeable solutions to their local problems. The distributed local search algorithm could be replaced by another one if sufficient information can be gathered regarding the relative level of difficulty of the complex local problems. The overall distributed systematic search algorithm could be replaced with any other complete algorithm which uses complex variables and dynamically re-orders its agents.

In summary, we have presented Multi-HDCS, a novel hybrid approach for solving DisCSPs with (naturally distributed) complex local problems which combines both distributed and centralised algorithms as well as local search and systematic search. Empirical results suggest that the Multi-HDCS approach is competitive when compared to other algorithms.

REFERENCES

[1] F. Rossi, P. van Beek, and T. Walsh, *Handbook of Constraint Programming*. Elsevier, 2006. [Online]. Available: http://ai.uwaterloo.ca/ vanbeek/cphandbook.html

[2] M. Yokoo and K. Hirayama, "Algorithms for Distributed Constraint Satisfaction: A Review," *Autonomous Agents and Multi-Agent Systems*, vol. 3, no. 2, pp. 185–207, 2000. [Online]. Available: http://citeseer.ist.psu.edu/yokoo00algorithms.html

[3] M. Basharu, I. Arana, and H. Ahriz, "Solving DisCSPs with Penalty Driven Search," in *Proceedings of AAAI 2005 - The Twentieth National Conference of Artificial Intelligence*, Pittsburgh, Pennsylvania, July 2005, pp. 47–52. [Online]. Available: http://publicoutputs.rgu.ac.uk/CREDO/open/mainpublications.php?novell=mb

[4] K. Hirayama and M. Yokoo, "The distributed breakout algorithms," *Artificial Intelligence*, vol. 161, no. 1–2, pp. 89–115, January 2005.

[5] M. Yokoo and K. Hirayama, "Distributed Constraint Satisfaction Algorithm for Complex Local Problems." in *ICMAS*, 1998, pp. 372–379.

[6] K. Hirayama, M. Yokoo, and K. Sycara, "An Easy-Hard-Easy Cost Profile in Distributed Constraint Satisfaction," *Transactions of Information Processing Society of Japan*, vol. 45, pp. 2217–2225, 2004.

[7] M. Basharu, I. Arana, and H. Ahriz, "Solving Coarse-grained DisCSPs with Multi-DisPeL and DisBO-wd," in *Proceedings of 2007 IEEE/WIC/ACM International Conference on Intelligent Agent Technology*. IEEE Press, 2007, pp. 335–341.

[8] D. Burke, "Exploiting problem structure in distributed constraint optimisation with complex local problems," Ph.D. dissertation, National University of Ireland, Cork, 2008.

[9] D. Lee, I. Arana, H. Ahriz, and K.-Y. Hui, "Multi-hyb: A hybrid algorithm for solving DisCSPs with complex local problems," in *Proceedings of 2009 IEEE/WIC/ACM International Conference on Intelligent Agent Technology (IAT 2009)*, 2009, pp. 379–382.

[10] D. Lee, I. Arana, H. Ahriz, and K.-Y. Hui, "A Hybrid Approach to Distributed Constraint Satisfaction," in *Artificial Intelligence: Methodology, Systems,and Applications. 13th International Conference, AIMSA 2008 Proceedings*, D. Dochev, M. Pistore, and P. Traverso, Eds., Varna, Bulgaria, September 2008, pp. 375–379.

[11] A. Meisels, E. Kaplansky, I. Razgon, and R. Zivan, "Comparing Performance of Distributed Constraints Processing Algorithms," in *Proceedings of the AAMAS-2002 Workshop on Distributed Constraint Reasoning*, Bologna, July 2002, pp. 86–93. [Online]. Available: citeseer.ist.psu.edu/meisels02comparing.html

[12] I. Brito, "Distributed constraint satisfaction," Ph.D. dissertation, Institut d'Investigacio en Intel.ligencia Artificial Consejo Superior de Investigaciones Cientificas, 2007.

[13] R. Bejar, C. Domshlak, C. Fernandez, C. Gomes, B. Krishnamachari, B. Selman, and M. Valls, "Sensor networks and distributed CSP: communication, computation and complexity," *Artificial Intelligence*, vol. 161, pp. 117–147, 2005.