Application of Max-Sum Algorithm to Radar Coordination and Scheduling

Yoonheui Kim[1], Michael Krainin[2], and Victor Lesser[1]

[1] University of Massachusetts at Amherst, MA 01003, USA,
ykim@cs.umass.edu
[2] University of Washington, Seattle, WA 98195

Abstract. The Max-Sum algorithm is a constraint optimization algorithm using message passing that can be configured to work in an approximate mode. It has been shown that in this mode it compares well with other approximate distributed optimization approaches. We used the approximate Max-Sum algorithm to coordinate adaptive radars for real-time weather sensing. When applied in various settings of the system with different types weather phenomena scenarios and radar configurations focusing on the utilization of structures in a distributed clustered organization, the Max-Sum algorithm performs efficiently in terms of resource utilization and communication for most of them in comparison to a negotiation-based algorithm specifically designed for this domain structure. We also modified the Max-Sum algorithm to utilitze the domain structure and have found that this utilization further improves the algorithm performance. Finally, we show that the Max-Sum is robust to initial policies.

1 Introduction

The Max-Sum algorithm is a message-passing based algorithm that maximizes a utility function of a constraint graph and is proven to work well for solving a certain class of constraint optimization problems [1]. It is also suitable for a distributed setting as the algorithm computes an optimization function by communicating the partial results with direct neighbors. It has been proven to work well in an approximate mode where the solution is not guaranteed to be optimal due to cycles in the constraint graph.

This paper exlores the distributed Max-Sum algorithm for radar scheduling in the NetRad system [3], which is designed for detecting and monitoring hazardous weather phenomena in real time. The NetRad system at the highest level is organized as a collection of controllers, each responsible for scheduling a cluster of radars based on the evolving weather scenario. Each radar's scanning strategy can be dynamically configured every 60 seconds. The radars are geographically located to overlap so that multiple radars can concurrently scan phenomena to obtain dual-Dopplar velocity vector measurements where desired.

There is a limit in the amount of weather volume that each radar can scan per cycle of 60 seconds and the radars have to coordinate to choose a highest-utility

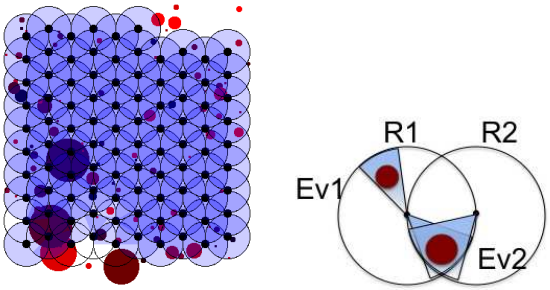
phenomena to scan each cycle so as to optimize the overall utility of the system. Modeling the coordination and scheduling of radars as a constraint optimization problem and applying the Max-Sum algorithm is suitable for this domain for two main reasons.

First, the Max-Sum algorithm can be used to generate a real-time radar scheduling policy. Since it quickly reaches a close-to-optimal solution exhibiting an anytime algorithm property. In this domain, weather phenomena change dynamically, and therefore the scanning strategy needs to be repeatedly recomputed.

Second, the radar scan scheduling problem can be naturally modeled as a constraint optimization problem. In the system, each weather phenomenon is regarded as a task with an associated utility. The quality of a radar scan depends on the scope of space covered of the weather phenomena. A single radar is often not able to scan all the phenomena in its range and thus benefits from collaborating with neighboring radars that can potentially scan the desired region. Coordination of radars are necessary due to pinpointing tasks in addition to the lack of resources e.g. time for scanning. Between two types of tasks, pinpointing and non-pinpointing tasks, pinpointing tasks require more than one radar to scan the phenomena, therefore tighter coordination among radars is necessary.

The main purpose of the work is to study the behavior of the Max-Sum algorithm in the problem domain so as to understand the performance of the Max-Sum algorithm in comparison to other algorithms including a previously developed negotiation algorithm [7] designed for the application. A key issue that we want to study is whether the optimization should be broken along the lines of the grouping of radars into clusters, which leads to a two-stage optimization process where we first optimize scheduling within a cluster and then modify that optimization based on interactions with other clusters. This two-stage process exploits the low communication overhead possible when optimization is done in the first stage. We contrast this with an approach that sees the optimization as a single integrated process that does not explicitly take advantage of the grouping. To understand this issue, we have tried evaluating the performance of two different algorithms: one which directly exploits the decomposed organizational structure and the other which does not. Additionally, we tried to improve the Max-Sum algorithm by providing a better starting point for the optimization process based on the results of a local optimization algorithm specific to each radar cluster.

The paper is organized as the following. In the next section, we introduce the NetRad system and formulate the constraint optimization problem we are trying to solve. The third section briefly overviews the Max-Sum algorithm and describes two approaches to formulate the optimization problem and also describes a modified Max-Sum algorithm which uses an initial policy. Then, we present the experimental results. Finally, we summarize the major conclusion of the paper and discuss future work.



(a) 96 radars with 96 phenomena

(b) 2 radars configurations

Fig. 1. Example weather scenario and radar settings (a), Example configuration of Radars (b). All radar ranges and phenomena are assumed circular shaped. All phenomena locations and sizes are randomly selected. In (b), Radar 1 (R1) can choose to scan event 1 (Ev1), event 2 (Ev2) or to scan both depending on the utility. Scanning all phenomena in range may not be possible given the time limit to scan.

2 NetRad System and Problem Formulation of Radar Scanning Policy

2.1 The NetRad system

The NetRad System is a system of radars specially designed for the purpose of quick detection of low-lying meteorological phenomena such as tornadoes. They are short-range radars used in dense networks, thereby alleviating blind-spots caused by the curvature of the Earth. NetRad radars additionally do not just use the traditional sit-and-spin strategy; rather, they can be focused to scan in a particular volume of space. By exploiting the collected information on weather phenomena, scanning strategies can be dynamically created for specific weather phenomena in the current environment.

The NetRad system consists of multiple MCCs (meteorological command and control) each of which controls a set of radars. The MCC system is a closed-loop control system in that it responds to the emerging weather events based on detected features in the radar data and end-user concerns that may vary over time. End-users such as forecasters, emergency managers, and researchers can provide information as to what sort of data they are looking for and how frequently. Consequently, the MCC ranks the importance of tasks so as to give preference to the data users want.

The MCC gathers moment data from the radars and then runs detection algorithms on this weather data. The result of this analysis leads to a set of weather-scanning tasks of interest for the next radar scanning cycle. The MCC then determines the best scanning strategy for the available radars that will maximize the sum of the utility associated with the chosen tasks according to a utility function based on the end-user priorities. This scan strategy is used by the NetRad radars on the next cycle.

Tasks may also be either pinpointing or non-pinpointing, meaning either there is, or is not, a significant gain by scanning the associated volume of space with multiple radars at once. The utility gained from scanning a pinpointing task increases with the number of radars scanning the task; the utility for a non-pinpointing task is the maximum utility among the individual radars that scan the same phenomenon. The global utility is simply the sum of utilities of all tasks. For a more in-depth description of the MCC system, see [2], or see [3] for more details on the utility function.

2.2 Problem Formulation of Generating Radar Scan Policy

The goal of the system is to maximize the overall utility by summing the utility calculated for each scanned task. Each radar can choose where to scan by choosing a subset of phenomena in its range. The possible scanning strategies of a radar are discretized. For each task that a radar can scan, there is a preferred scan: one which covers the task region most tightly. The preferred scan for each task is in the scan set as well as mergers of each pair of preferred scans into new, combined sweeps.

For each phenomenon t_i, the utility function

$$u_i : t_i \rightarrow r \in \Re, \text{ where } 0 \leq r \leq 1 \tag{1}$$

is determined by the priority of the requesting user or the weather pattern.

Also, there is a function for the quality of scan for each phenomenon t_i

$$q_i : t_i \times (r_1, \ldots, r_n) \rightarrow r \in \Re, \text{ where } r_j \text{ denotes the scanning policy of radar } j. \tag{2}$$

The radars in the quality function argument are limited to the radars which have the phenomena in range.

The system utility U is defined as

$$U = \sum_i u_i \times q_i. \tag{3}$$

and thus the goal of the sytem at each scan cycle is to find the configuration of radars $r_1, \ldots, r_n$ which maximizes the system utility U. This problem easily maps into a constraint optimization problem where there is a variable corresponding to each radar whose value specifies which of the possible sweeps the radar should execute on the next scan cycle.

3 Max-Sum algorithm and Modeling on Radar Scanning Problem

3.1 The Max-Sum algorithm

Max-Sum is a distributed message-passing optimization algorithm belonging to the class known as Generalized Distributive Law (GDL) [4]. Max-Sum is a variation of Sum-Product algorithm but tries to maximize the global utility function.

In the Max-Sum algorithm, there is a set of variables $\mathbf{x} = \{x_1, x_2, \ldots, x_m\}$ on which a set of functions $\mathbf{F} = \{F_1, F_2, \ldots, F_n\}$ depend. Each function $F_i = F_i(\mathbf{x}_i)$, $\mathbf{x}_i \subset \mathbf{x}$. The goal is to find $\mathbf{x}^*$ which satisfies the following:

$$\mathbf{x}^* = \arg\max_{\mathbf{x}} \sum_{i=1}^{n} F_i(\mathbf{x}_i) \tag{4}$$

Therefore, the Max-Sum algorithm can be viewed as a constraint optimization algorithm and we are looking for the settings of variables which maximize the sum of a set of local utility functions. To achieve this, the Max-Sum algorithm defines a factor graph by creating a node for each variable and for each function. The graph is bipartite, and a function node is connected to a variable node if the corresponding function is dependent upon that variable. The bulk of the algorithm is in the messages passed between nodes, which are:

Variable i to Function j:

$$q_{i \to j}(x_i) = \alpha_{ij} + \sum_{k \in M_i \setminus j} r_{k \to i}(x_i) \tag{5}$$

Here α_{ij} is a scalar set such that $\sum_{x_i} q_{i \to j}(x_i) = 0$, and M_i contains the indices of function nodes connected to variable node i.

Function j to Variable i:

$$r_{j \to i}(x_i) = \max_{\mathbf{x}_j \setminus i} [F_j(\mathbf{x}_j) + \sum_{k \in N_j \setminus i} q_{k \to j}(x_k)] \tag{6}$$

where N_j contains the indices of variable nodes connected to function node j in the factor graph.

If the factor graph is cycle-free, then the messages are guaranteed to converge, and the resulting solution will maximize $\sum_{i=1}^{n} F_i(\mathbf{x}_i)$. When the graph contains cycles, the messages may not converge, and even if they do the resulting solution may be sub-optimal. Empirical results show that even in this case, the algorithm frequently provides good solutions [1].

3.2 Modeling Max-Sum in the NetRad system

We explored two formalizations to implement Max-Sum in the NetRad system. The first of two formalizations (which we call coarse-grained) is based on the one proposed in [5] where a variable and a function node is created for each MCC. The variable node represents the joint scanning strategy of all radars controlled by the MCC and the domain size of this single variable is quite large. The function node represents the utility for all tasks which can be performed by the MCC. A function node is connected to a variable node if the function could conceivably depend on the variable, that is, if the MCCs are the same or neighbor each other. As some of the tasks are shared by multiple MCCs, each function node considers the same tasks multiple times, so we divide the utility of a task by the number of function nodes counting it. The function node considers

configurations of directly connected radars in the neighboring MCCs to minimize the size of configurations considered in each function node.

The second formalization (which we will call fine-grained) disregards the MCC organization and works at a finer granularity of individual radar rather than at the MCC level. Each radar is regarded as its own variable whose domain is the set of allowable scans it may perform. Each task is its own function node whose value is the single-task utility discussed in Section 2. A function node is connected to a variable node if the radar is capable of scanning the weather phenomenon associated to the task.

The two formalizations have their own advantages. The factor graph for the first formalization is independent of the number of tasks in the system and even to some extent how the radars are assigned to MCCs. Additional tasks only increase the size of the variable domains in the first formalization, not the number of function nodes to which variable nodes are attached.

The first formalization has much bigger domains for the values each variable and function node can take. Without well-designed heuristics, the computational complexity is too high to be applied directly. With 10 values for each variable nodes, the common size of the domain for function nodes is 12^{10} where 4 radars belong to one MCC (configuration of 4 radars in an MCC and 8 neighboring radars). This makes the straightforward application of the first formalization undesirable in the on-line settings and a heuristic to cut down the complexity should be used. Therefore, we mainly experimented with the fine-grained formulation of the Max-Sum algorithm for this application domain.

3.3 The Max-Sum algorithm with Initial Policy

In the Max-Sum algorithm, a node's outgoing messages are dependent upon the incoming messages it received in the prior cycles. At the start of the algorithm, no messages have been sent, so certain initial values must be used. The obvious choice is to set all values for messages not received to zero. In this way, initial variable node messages are uniform functions, indicating no preference for any specific variable assignment. A function node j's initial message to a variable node i indicates only the maximum local utility F_j given x_i.

If we were to stop here, each variable i would take on the value

$$\tilde{x}_i = \arg\max_{x_i} \sum_{j \in N_i} \max_{\mathbf{x}_j \setminus i} F_j(\mathbf{x}_j) \tag{7}$$

In other words, each variable would set itself in order to maximize the local utilities of neighboring functions assuming those functions get best-case settings of other variables for the local utility. This initial state is not only determined by a fairly local optimization, but it does so by assuming maximizations which may be neither mutually compatible nor even close to optimal for other parts of the factor graph.

As the algorithm proceeds, more global information begins to become incorporated in the messages; however, it is unclear whether encoding more global

information in the initial messages might lead to convergence to better solutions, faster convergence, or even convergence when the algorithm would otherwise fail to do so in loopy factor graphs.

We would like to be able to take some global variable assignment produced by some optimization technique and introduce it somehow as a starting configuration to replace that in Equation 7. To do so, we modified the algorithm to always start with function nodes sending messages at the beginning of the algorithm.

Once the optimization algorithm of choice has been used to construct a variable assignment $\hat{\mathbf{x}}$, function nodes send the following messages to the connected variable nodes. Here j is the index of the function node and i that of the variable node.

$$r_{j \to i}(x_i) = F_j((\hat{\mathbf{x}}_j \setminus i) \cup x_i) \tag{8}$$

This is the value of F_j when i has state x_i and all other variables use their states from the solution $\hat{\mathbf{x}}$.

After receiving these messages, if a variable node were to take on a value, it would be:

$$\tilde{x}_i = \arg\max_{x_i} \sum_{j \in N_i} F_j((\hat{\mathbf{x}}_j \setminus i) \cup x_i) \tag{9}$$

Proposition 1 *If the assignment $\hat{\mathbf{x}}$ is such that no* individual *variable can by itself change its value to increase the global utility, then $\hat{\mathbf{x}}$ is a solution to the assignment constraints imposed by Equation 9. If changing any* individual *variable's value will strictly decrease the global utility, then $\hat{\mathbf{x}}$ is the unique solution for Equation 9.*

Proof. From the perspective of an arbitrary variable node i, all other nodes are fixed to the configuration specified by $\hat{\mathbf{x}}$. Maximizing $\sum_{j \in N_i} F_j((\hat{\mathbf{x}}_j \setminus i) \cup x_i)$ leads to maximize the global utility given the values of other variables. This is because only the functions for nodes $j \in N_i$ are affected by x_i.

If $\hat{x}_i$ were not a solution to this, then the algorithm which selected $\hat{x}_i$ to be part of $\hat{\mathbf{x}}$ could have instead selected $\tilde{x}_i$ to receive a higher utility. Since by supposition, no individual variable can change its value to increase the global utility, $\hat{x}_i$ is a solution to Equation 9. If changing any variable's value in $\hat{\mathbf{x}}$ will decrease the global utility, then there can be only one solution to Equation 9. Since $\hat{x}_i$ is a solution, it must be the unique solution.

Thus, in the sense of the above property, we can insert a variable assignment into a factor graph as a starting solution. The property requires that no single variable can change its value to increase the utility. This is a desirable property for an optimization algorithm to have, and a fairly lax one. Any algorithm which does not satisfy this constraint can be followed by a hill-climbing procedure in order to meet the requirement of Property 1.

In addition to what Property 1 can tell us, Equation 9 by itself looks quite a bit better than Equation 7. While the assignment still only considers directly neighboring function nodes, it does so using better assumptions. For nodes other

than itself, it assumes a configuration that is known to exist rather than a separate maximization for each function node. The assumed variable assignments are also known to be consistent with a good global utility, and x_i will fit itself into this assignment.

After the messages from function nodes to variable nodes, we allow the variable nodes to send one set of messages before proceeding with the regular algorithm. This is so the next set of messages from function nodes will have a starting point other than assuming uniform functions in variable node messages.

In this paper, we focus on the performance of Max-Sum for a single snapshot of weather and do not deal with dynamics of system; it is interesting to know how in a dynamic setting the previous result could be used in the next cycle.

4 Experimental Results on the Performance of Max-Sum on NetRad

We experimented with the Max-Sum algorithm described in Section 3 in various settings. The purpose of our experiments were twofolds. We are particularly interested in understaing how the algorithms perform in the cluster-organizational setting where the whole network of radars is decomposed into several clusters and each cluster is managed by an MCC. We also ran tests for the Max-Sum algorithm with an initial policy and studied how the initial policy affects performance.

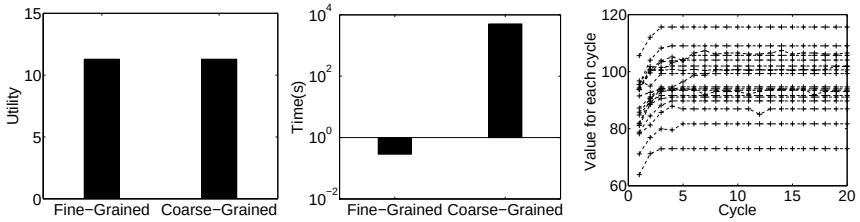
4.1 Experimental Setting

To run our experiments, we used an abstract simulation environment of the NetRad radar system developed in the Farm simulator framework [6]. In this simulator, weather tasks are abstracted as circular areas as shown in Figure 1(a), and time is discretized into system heartbeats. Aspects such as the utility function, the effective range of radars, and the separation between radars, however, are the same as in the operational testbed. For more information on the simulation environment, see [7].

We compared the results of several optimization techniques over a number of trials; what varies is the number of radars, number of weather phenomena in the environment and spatial extent of weather phenomena. To make the results more easily interpretable, each trial is run for only one scan cycle. By only being concerned with one scan cycle in isolation, the difference among approaches can be seen more clearly; if a trial is involved multiple scan cycles the result of previous scan cycles decisions will affect the set of tasks available in the next scan cycle; thus making a direct comparison among approaches more difficult.

4.2 Performance of Max-Sum in Two Modeling

We first experimented with the two modelings of Max-Sum described in Section 3.2 on the 12-radars network. The problem with 12 radars is trivial in a sense



(a) Performance Quality (b) Time complexity (c) Value Trend

Fig. 2. The performance (a) and time complexity (b) of two Max-Sum modelings on 12-radar network with 12 weather phenomena. Both modelings are able to solve the problem with a similar quality. However the local optimization step in function nodes in coarse-grained modeling takes too much time, showing that the fine-grained modeling exploits the simplicity of the Max-Sum algorithm. (c) Value trend of Max-Sum over multiple cycles in 48 radar network. Max-Sum shows convergence and anytime algorithm characteristics by converging to a final value within 5 cycles for most cases and the result of first cycle being also close to the final solution.

that different algorithms found the same solution with the same utility, but the coarse-grained version takes too much time because of the computation in large function nodes. The straightforward application to the cluster-organizational structure in the coarse-grained model leads to difficult subproblems with large domains in function nodes. This is a known problem with the Max-Sum algorithm: the complexity of function nodes increases exponentially very easily with the number and domain size of connected variable nodes and heuristics should be used to reduce the complexity. It is better to avoid such exponential explosion. Although the coarse-grained model is formulated along the decomposed models with two steps of local and global optimization aligning the system structure, the model does not exploit the full benefit of simple local computation of Max-Sum whereas the fine-grained formulation does.

In addition, the performance of the fine-grained Max-Sum algorithm in the 48-radar setting shows its anytime property where it converges to a close to optimal solution very quickly.

4.3 Performance of the Max-Sum algorithm on NetRad System in Comparison to Other Algorithms

In this section, we evaluate the performance of several alternative optimization algorithms, while varying the number of radars and the number of phenomena to show the performance of the Max-Sum algorithm in the domain with the cluster-organizational structure. We compare the performance of fine-grained implementation of Max-Sum decentralized message-passing algorithm to a decentralized negotiation algorithm [7] and a centralized optimization algorithm based on a genetic algorithm that is currently used for local optimization in the negotiation algorithm in each MCC.

The negotiation algorithm, specifically developed for the NetRad problem domain, is an iterative two step process perfromed concurrently at each MCC. In the first step each MCC performs a local optimization based on its local tasks

and knowledge of how neighboring MCCs scan schedules are related to its local tasks. In the second step, the MCC negotiates with their neighbors so as to make adjustments to their scheduling based on the strategy of other MCCs. This two step process for performing the distributed optimization tries to maximize the parallelism at the MCC level and to minimize communication among MCCs.

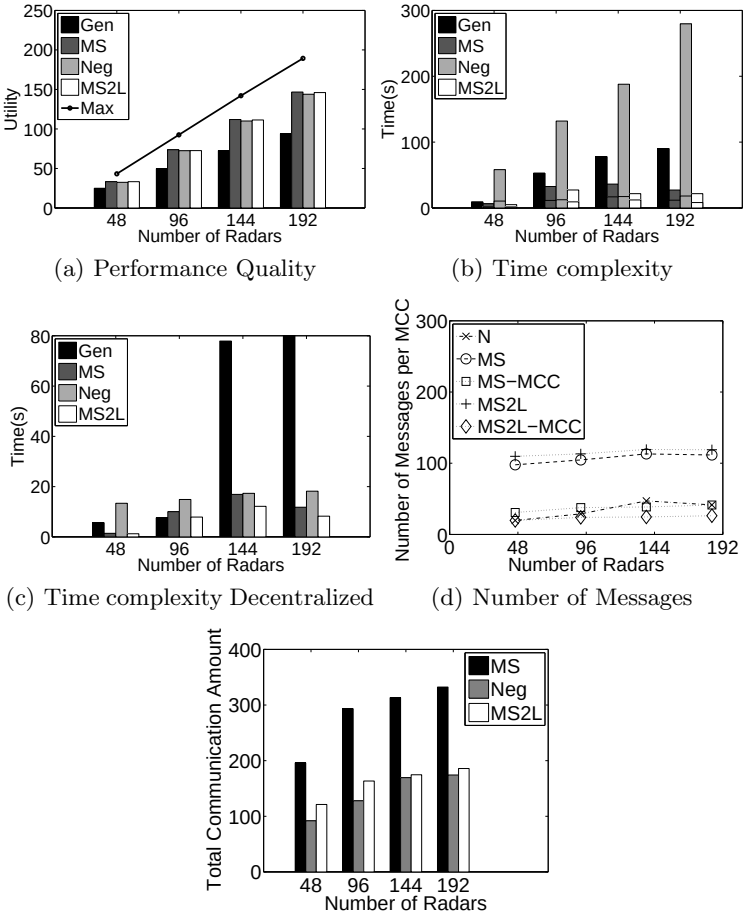
In contrast, the Max-Sum algorithm does not consider such an organizational structure and is completely decentralized. The Max-Sum algorithm does not explicitly take into account that certain links are within an MCC cluster and others are between MCC clusters. The genetic algorithm uses a centralized approach; no communication is required and it only utilizes one processor.

Performance for Different Network Sizes In order to evaluate the general performance and the scalability of the algorithms, we compare the performance on different sized networks. In the scenarios, there are the same number of phenomena as the number of radars in the network; the size and location of the phenomena are randomly chosen. The result is shown in Figure 3. The performance quality of Negotiation and Max-Sum is close for all network sizes whereas the performance of the centralized genetic algorithm is always inferior to these algorithms. This result show that the Max-Sum algorithm is able to handle the problem well without explicitly exploiting the clustering of radar controllers.

Each bar in Figure 3(b) shows the total time taken to run the algorithm in a single processor and shows that the Max-Sum algorithm quickly converges to a solution as good as the result of Negotiation algorithm. However, the result does not consider any benefit of decentralized computation of both algorithms and the estimated time for the decentralized setting is given as well.

The lower stacks in Figure 3(b) represents the estimated computation time when the algorithms are run in the decentralized setting. The estimated time is computed as the sum of the longest local computation time for each round. In both negotiation algorithm and the Max-Sum algorithm, each MCC, when the local computation is done, waits for other MCCs to finish the computation and then they exchange messages. Therefore, the time complexity in the decentralized setting results from the sum of the longest time taken in the local computation for each round. The Max-Sum algorithm benefits in terms of decreased computational time and higher quality from the conceptually more decentralized computational structure though it pays some cost in terms of additional communication among MCCs and still have a comparable performance to the negotiation algorithm.

In addtion, in order to assess the communication burden of Max-Sum, we measured the number of messages exchanged across MCCs and compare it with that of negotiation algorithm as in Figure 3(d). When only messages exchanged across MCCs are counted, Max-Sum needs more than twice the communication than the negotiation algorithm. In Section 4.4, we will discuss a variant of the Max-Sum algorithm(MS2L) that has comparable communication efficiency as the negotiation algorithm without sacrificing its computational efficiency.



(a) Performance Quality

(b) Time complexity

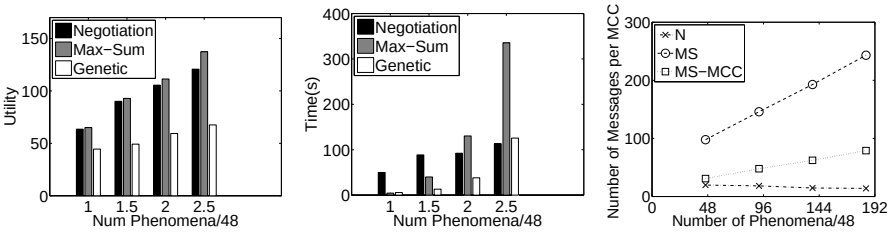
(c) Time complexity Decentralized

(d) Number of Messages

(e) Total Amount of Communication

Fig. 3. Gen:Genetic MS:Max-Sum Neg:Negotiaton MS2L:Max-Sum-2-Level MS-MCC:Max-Sum across MCCs only, and MS2L-MCC:Max-Sum-2-Level across MCCs only. The performance (a) and time complexity (b) and estimated computation time in a decentralized setting(c) of algorithms on different sizes of the network given the same number of tasks (weather phenomena) as the number of radars. The lower stacks in (b) represent the estimated computation time in a decentralized setting which are also separately shown in (c). It shows that the fine-grained version of the Max-Sum algorithm is able to solve the problem with a similar quality to the negotiation algorithm more quickly. (d) The number of messages of negotiation and Max-Sum per MCCs. Max-Sum across MCCs only represents the number of messages sent across MCCs in Max-Sum. (e) Total amount of communication in terms of the average size of messages $\times$ the number of messages. Message size is measured in terms of number of utility values in the messages.

Performance varying with the ratio of the number of phenomena to the number of radars In the next experiment, we increase the number of phenomena in a 48-radar network, thereby requiring more coordination between radars and studied how the algorithms perform. While the quality of solution of Max-Sum is slightly better, the time complexity of the Max-Sum algorithm sharply increases because the number of function nodes in Max-Sum increases as more weather phenoemena are added.



(a) Performance Quality (b) Time Complexity (c) Number of Messages

Fig. 4. The performance quality (a), time complexity (b) and number of messages (c) of algorithms on different number of phenomena. The basis is 48 weather phenomena and this is increased to 120 phenomena.

When the number of phenomena increases, the maximum number of variable nodes per each function nodes increases. This leads to a computational complexity hike in Max-Sum as shown in Figure 4(b). Also, the number of messages across MCCs increases as there are more tasks shared by multiple MCCs in the environment. In contrast, number of messages in the negotiation algorithm decreases due to the failed negotiations resulting in early termination. It is observed throughout the experiments that the negotiation algorithm terminates early with a suboptimal solution after one or two negotiations when the problem gets difficult as shown in this experiment.

4.4 Performance of Max-Sum in a Two-Level Hiearchy

In the previous results, the fine-grained version of the Max-Sum algorithm shows good performance in the domain without exploiting the cluster-organizational structure. We then modified the algorithm to exploit this system structure and experimented on the Max-Sum algorithm with a two-level hiearchy. That is, we ran the Max-Sum algorithm on the MCC-level factor graphs consisting of only the nodes in each MCC for three rounds and then optimized on the global-level factor graph as normal.

The result in Figure 3 shows that Max-Sum in two-level actually beats the performance of the regular Max-Sum algorithm in every aspect we experimented on. The performance quality remains similar to Max-Sum and the time complexity decreases. The computation on the factor graph with local nodes only is much simpler than on the global-level factor graph and also the result of this computation leads to a quicker convergence on the global level.

As messages are exchanged only within MCCs for three rounds, the number and size of messages also decreases. The result in Figure 3(d) shows that the number of messages of Max-Sum in two level is smaller than negotiation algorithm. This result indicates that in this domain Max-Sum does not necessarily need to exploit the cluster-organizational setting, but the use of the setting benefits the algorithm.

4.5 Performance using Initial Policy

As a final set of experiments, here we present the result on an extension of the Max-Sum algorithm described in Section 3.3. As in Section 3.3 we tried to augment the algorithm with an initial policy for guiding the optimization process. For testing the algorithm with an initial policy, we tested two kinds of initial policy, one random and the other generated by a centralized optimization - a genetic algorithm. As explained, this initial policy is given to function nodes, which send messages that lead the variable nodes to have preference for a scan policy maximizing the utility given the initial policy of other variable nodes.

To show the effect of the initial policy on the Max-Sum algorithm, we started the algorithm with initial random policies for two specific weather scenarios. As in Figure 5(a), the final solution quality does not vary even with different initial policies. This shows that the Max-Sum algorithm is resilient to initial policies even with cycles in the factor graph.

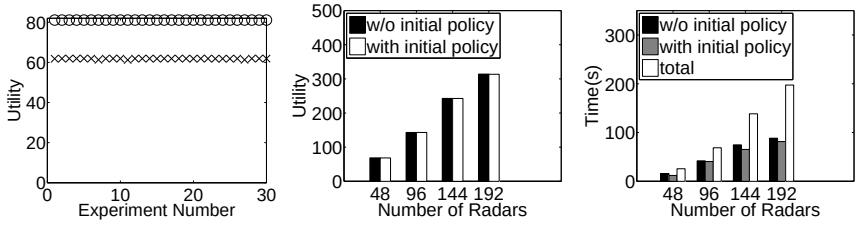


Fig. 5. (a) The performance quality of Max-Sum with randomly generated initial policy given two fixed phenomena (one symbol for each phenomenon). (b) The performance of Max-Sum with initial policy using a result from the genetic algorithm (c) Time complexity of Max-Sum with initial policy using a result from the genetic algorithm. total denotes the time for both caculating the initial policy with genetic algorithm and calculating Max-Sum.

Additionally, we initialized the Max-Sum algorithm with a better initial policy since randomly generated initial policy can be very ineffective. We took the solutions from the genetic algorithm for each MCC whose value is suboptimal in comparison to other algorithms but generally better than random policies and give them as initial configurations. As shown in Figure 5, the performance does not improve except for a reduction in time. However, including the time taken to compute the policy negates the gain in time. This result is interesting in that

Max-Sum is very resilient to the initial policy and even when it is given a skewed value during the computation, the algorithm still converges to a similar solution.

5 Conclusion

The Max-Sum algorithm is an approximate constraint optimization algorithm using message passing. We applied the algorithm in the NetRad system for coordinating weather-sensing radars. In this system, fine-grained modeling of Max-Sum worked well even without utilizing the cluster-organizational structure while not requiring much more communication than negotiation algorithm. Also, the coarse-grained modeling of Max-Sum has shown the limitation of utilizing the decomposed structure as the local computational burden is too high. We also applied Max-Sum in the two-level hierarchy which works with the local nodes within the MCC at the beginning and later expand the scope to the whole network. This version of the Max-Sum algorithm proved the benefit of using the organizational structure, spending less computation time and less number of messages than all other algorithms. Finally, we applied initial policies designed to improve the algorithm. However, the Max-Sum algorithm showed that it is very resilient to a transient preference towards a given configuration.

The Max-Sum algorithm has a well-known limitation of computational explosion for function nodes and our experiment with the fine-grained version also suffers from such explosion when the the number of phenomena and radar radius are increased. This work on the Max-Sum algorithm suggests some directions for future research. One is to study ways for decomposing the function nodes connected to many variable nodes. In a series of experiments on various settings not shown in this paper due to space limitations, the algorithm suffer from computation explosion as the number of phenomena and radar radius increases. Although many heuristics were proposed to guide the search in the function node itself, it would be beneficial to minimize the size of the function nodes by decomposing the search space into smaller ones. For instance, the function nodes in the domain can be decomposed into a set of smaller function nodes for each radar for non-pinpointing tasks as the final utility of such tasks comes entirely from a single radar.

Another possible direction is a decomposition of factor graph where the connection is loose. Temporarily restricting the connections to a subset of variable nodes based on the factor graph will greatly reduce the complexity of the function nodes. This will also reduce the communication burden given on the Max-Sum algorithm as well as the computational complexity.

Finally, as shown in the experiment on initial policies, Max-Sum is very resilient to a transient preference for a specific configuration. This might also mean that in a dynamic environment when tasks are changing, Max-Sum could adapt to the change well as it can find the same final solution from any starting point. Also, in a sense that the problem is loosely connected across the whole network, a local change in the problem would not affect the solution globally and only partial solution should be recomputed.

Acknowledgement

This work was supported in part by the Engineering Research Centers Program of the National Science Foundation under NSF Cooperative Agreement No. EEC-0313747. Any Opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect those of the National Science Foundation.

We would like to thank Alessandro Farinelli, Alex Rogers, and Ruben Stranders of the University of Southampton for their assistance in this project. Their suggestions such as branch and bound search helped us to efficiently implement the Max-Sum algorithm. They also suggested the fine-grained formalization as an alternative to our initial approach.

References

1. Ruben Stranders, Alessandro Farinelli, Alex Rogers, and Nick Jennings. Decentralised coordination of mobile sensors using the max-sum algorithm. In *Proc 21st Int. Joint Conf on AI (IJCAI)*, 2009.
2. M. Zink, D. Westbrook, S. Abdallah, B. Horling, E. Lyons, V. Lakamraju, V. Manfredi, J. Kurose, and K. Hondl. Meteorological Command and Control: An End-to-end Architecture for a Hazardous Weather Detection Sensor Network. In *Proc. of the ACM Workshop on End-to-End, Sense-and-Respond Systems, Applications, and Services*, pages 37–42, 2005.
3. James F. Kurose, Eric Lyons, David McLaughlin, David Pepyne, Brenda Philips, David Westbrook, and Michael Zink. An End-User-Responsive Sensor Network Architecture for Hazardous Weather Detection, Prediction and Response. In *Proceedings of the Second Asian Internet Engineering Conference, AINTEC*, pages 1–15, 2006.
4. S.M. Aji and R.J. McEliece. The generalized distributive law. *Information Theory, IEEE Transactions on*, 46(2):325–343, Mar 2000.
5. Alessandro Farinelli, Alex Rogers, and Nick Jennings. Maximising sensor network efficiency through agent-based coordination of sense/sleep schedules. In *Workshop on Energy in Wireless Sensor Networks in conjuction with DCOSS 2008*, pages IV–43–IV–56, June 2008.
6. Bryan Horling, Roger Mailler, and Victor Lesser. Farm: A Scalable Environment for Multi-Agent Development and Evaluation. In *Advances in Software Engineering for Multi-Agent Systems*, pages 220–237. 2004.
7. Michael Krainin, Bo An, and Victor Lesser. An Application of Automated Negotiation to Distributed Task Allocation. In *2007 IEEE/WIC/ACM International Conference on Intelligent Agent Technology (IAT 2007)*, pages 138–145, Fremont, California, November 2007. IEEE Computer Society Press.