

# A hierarchical model for processing noisy and partial information in large-scale real-time task environments

**Keywords:** Multi-Agent Systems; Real-time Systems

**Tracking number:** E0309

**Abstract.** In this paper we introduce the Incremental Distributed Dispatcher Manager (IDDM) that is designed to handle control problems with large numbers of tasks and cooperative agents where only partial and noisy information is available. The IDDM is a modification of the DDM model that was developed to solve similar problems but in environments where accurate information is available. There were a number of challenges that had to be addressed in developing the IDDM: (1) agents must be able to process noisy information which they then use to compute a partial solution to a local task; (2) the system must be able to integrate the partial results obtained by several agents into a more accurate global solution; and (3) even though the information may be extremely noisy, solutions should be developed and integrated in real time. Our approach for handling noisy information is to first estimate which information is incorrect, and then to use only the remaining information to form partial solutions. We tested our model and algorithms in an environment of many mobile Doppler sensors and targets. Our experiments demonstrate that our agents reach a high level of task satisfaction even with a high rate of noise.

## 1 INTRODUCTION

The Incremental Distributed Dispatcher Manager (IDDM) is a hierarchical model that manages large-scale agent systems in noisy environments. The environments are large (virtual) areas occupied by a large number of tasks and a large number of task solving agents. In a large-scale environment it is usually necessary to use many simple and cheap agents instead of fewer sophisticated but expensive ones. However, measurements taken by such simple agents usually are less accurate and the agents may need to use partial and noisy information when trying to solve tasks. Thus, handling partial and noisy information is essential in large-scale agent systems. In previous work similar environments with partial information characteristics but without noise were considered [11]. For those environments, a hierarchical model and a mechanism to combine partial results to solve tasks was proposed. In this paper we present a modified version of that model that is appropriate for handling incorrect information by following a data reusability strategy. A set of comprehensive experiments was conducted to test the IDDM performance in such environments. There are several domains where such problems arise: satellites that are tasked to form a general picture of a large area; satellites that form weather maps; agents that control air or ocean pollution; sensor webs that watch geographic areas

for passing aircraft; and unmanned air and ground vehicles that must be jointly tasked on surveillance missions.

We applied the IDDM model to a challenge problem from the DARPA Autonomous Negotiation Teams (ANTS) program and developed a simulation to test the model. In the ANTS problem sensing agents must be allocated to moving targets (tasks) in a distributed and a real time environment.

## 2 ENVIRONMENT DESCRIPTION

We consider environments satisfying the following properties:

1. There are many task-entities in the environment (possibly virtual).
2. Each task-entity is associated with a task-state at a given time and the state of a task-entity may change over time. We denote the set of all the task-states  $TS$ .
3. There is a set of Sampling Agents (SA); each is able to take measurements. Each  $sa \in SA$  is associated with an agent state that may change over time. We denote the set of all possible sampler-agent states  $SAS$ .
4. Sampling-agents are able to obtain measurements on the task-entities if they are (virtually) located near the task-entities. The measurements provide only partial information on the task-states and may be mistaken. The set of all possible measurements is denoted  $MTS$ .
5. Given  $k$  correct consecutive measurements taken by the same agent, ending at time  $t$ , there is a function  $PosTS$  that returns a set of possible states for the task-entity at time  $t$ . If at least one of the measurements is incorrect, the function will fail.
6. Given two task-states  $s1$  and  $s2$  and two time points  $t1$  and  $t2$ ,  $t2 > t1$ , there is a Boolean function  $ResBy$  that returns true if it is possible that if the task's state was  $s1$  at  $t1$ , it could be  $s2$  at  $t2$ .

Note that the  $ResBy$  is applicable only for correct samples. In this paper we consider cases of noisy measurements. Even a small deviation in one of the samples prevents the agent from computing possible states. Furthermore, in such situations the  $PosTS$  function fails, but does not identify the specific measurement that caused the failure. This means that the agents can detect that among  $N$  measurements there is one that is incorrect, however, it cannot determine which measurement out of the  $N$  consecutive samples is the noisy one. Therefore, in a noisy environment one incorrect measurement will cause one to reject all  $N$  consecutive measurements. We have improved the basic DDM model to overcome this problem. In the improved

model, noise in one of the measurements will not cause the rejection of all consecutive  $N$  measurements but will allow the formation of a new set of  $N$  consecutive measurements out of the accurate measurements. We achieved this improvement by developing methods for detecting and isolating the noisy measurements. Improving DDM so that it would make use of only what were believed to be the most accurate measurements also increases the fault tolerance of the model. Alongside the benefits of reusing accurate measurements we found that a disadvantage of the IDDM model was the need for more computation time to handle the same number of measurements. However, if the complexity of the PosTS is low enough, as in our ANTS application, the agents are still able to find solutions in real time.

### 3 THE INCREMENTAL DDM MODEL

The organization structure of the IDDM is similar to that of the DDM. We recall here the main ideas from [11]. The centers in both models are organized hierarchically and each is associated with a particular virtual zone. We refer to each center as a coalition leader, cl. Agents report to an immediate cl that passes the information to its leader. Each cl passes down instructions to direct agents.

A fixed number of agents are available to sample task entities in the controlled virtual zone. The agents use an internal clock to tag each measurement with a time. The clocks in the system need only be synchronized occasionally; although this can result in small differences between the clocks, the system can deal adequately with such variations.

Upon retrieving measurements the sampling agent computes the possible states for each sampled task entity. Task states are produced using  $N$  correct consecutive measurements.

**Definition 1:** *CMTS* is a set of vectors. Each vector consists of  $N$  consecutive measurements taken by the same  $sa \in SA$ . Formally,  $CMTS = \{ \langle t_0, m_0 \rangle, \dots, \langle t_{N-1}, m_{N-1} \rangle \}$

$m_i \in MTS$  and taken by the same agent,  $t_{i+1} = t_i + 1$

Every sampled task entity may be associated with several possible *task states* derived from the same raw data. All possible *task states* are combined with the sampling agent state to a capsule.

**Definition 2:** A *capsule* is a pair of task-states set and a sampling agent state.

$capsule = \langle sa, \{ts_0, \dots, ts_{N-1}\} \rangle$  where  $sa \in SA, ts_i \in TS$

Given a sampling agent state, a capsule represents a few possible states of a task entity at time  $t$  as measured by the capsules are the basic information unit communicated by sampling agents to their cl up the hierarchy to the top.

The IDDM model includes cl agents. The cl agents control virtual zones. Each higher level controls a larger virtual zone. We will refer the controlled virtual zone as a *zone*. We distinguish between two types of cl agents: a *zone coalition leader* agent, zcl, and a *sampler coalition leader* agent, scl. While the zcl controls other cl agents the scl controls the

behavior of a set of sampling agents in its controlled zone. The main goal of both types is to obtain information about task entities in its controlled area. Another important role of the zcl is to balance the number of sampling agents in its zone. The main difference between a zcl agent and a scl agent is that the first is responsible for an area combined out of different zones while the latter is directly responsible for the behavior of the sampling agents in a specific zone. Therefore, many of the zcl agents may be in charge of other zcl agents at a lower level. The zcl agents are in turn in charge of scl agents. The top-level cl processes the up coming capsules to form a global information map.

**Definition 3:** An *information map* is a set of pairs  $\langle p_i, ts_i \rangle$  where  $ts_i$  is a function from time periods to TS and  $0 < p_i \leq 1$ .

Intuitively, each such pair indicates that the sampling agents estimate that, with probability  $p_i$  there is a task-entity in the environment and that the way its state is changing over time is specified by  $ts_i$ , i.e., the state of this state-entity at a given time  $t_j$  is  $ts(t_j)$ .

We do not distinguish between the two types of cl agents when we apply the algorithm of information map formation in the top-level cl. However, the control algorithms for zcl agents and sampler coalition are different. Whereas the zcl should balance the number of sampling agents between zones and should decide how many should pass from one zone to another, the scl should follow orders from its superior zcl and decide which sampling agent to pass and how to do this. scl should also direct the sampling agents in the zone that it is in charge of. We will not discuss the latter sampler role further in this paper.

The IDDM design has important fault tolerant properties. A sampling agent can calculate possible states of a task without the need of another agent. A sampling agent may also acquire information about the same task from a different point of view and establish the right description of the task entity as a function of time. As we use more sampling agents we gain better performance. In addition if one of the leaders is disabled, it is replaced by one of the other agents below it by its leader. If the top-level leader stops functioning, then the leaders at the second level will distributively choose a top-level cl replacement. These properties are inherited from DMM. However, the main advantage of IDDM is its ability to handle extremely noisy information.

We can apply IDDM to many problems by mapping the IDDM entities to the domain entities. For example, in the case of satellites used to track forest preservation we can map each sampling agent to a satellite and each forest segment under surveillance to a task entity. In a later section, we will discuss an implementation of the IDDM in the ANTS domain involving mobile targets that are to be tracked by an array of Dopplers.

### 4 ALGORITHM DESCRIPTION

The IDDM uses the three algorithms of the DDM model alongside with a general algorithm for cmts generation that will be presented below for the IDDM.

Each sampling agent will apply the cmts generation algorithm to produce a new cmts to the capsule generation algorithm. The capsule generation algorithm calls the domain dependent function, PosTS, to deduce a set of possible task states using the generated cmts. It uses the set of possible task states and the sampling agent state to produce a capsule.

The IDDM top-level cl adopts two more algorithms from DDM. The first is responsible for combining incoming capsules to the global information map using the domain dependent function PosTS. The inputs to this algorithm are the sampling agent state and the cmts generated by an IDDM function. We described below how IDDM generates a cmts. The second algorithm is responsible for generating the information map.

We present here the algorithm to be used by a sampling agent upon receiving new noisy measurements. The algorithm is responsible for the generation of one cmts. The algorithm takes the newest N-1 measurements of the last cycle alongside with an incoming measurement and tries to form new cmts. In the case of one measurement damaged due to sampling noise no cmts will be produced. Reusing accurate samples can overcome such noise. Adopting the IDDM produces more cmts than the basic DDM. Calculating tasks states out of the N cmts is a difficult task therefore; in a real time environment the processing time may be a crucial factor.

#### Cmts generation algorithm

**Input:** former list of measurements (*old\_list*), a new measurement (*s*)

**Output:** new list of measurements (*new\_list*), a cmts (cmts)

*new\_list* = *old\_list*

// create new empty list if there was no previous information or if the new measurement is not consecutive to the last one store in the old list

if (*new\_list* == null || *s.time* != *new\_list.last().time*+1)

*new\_list* = new List()

*new\_list.addToTheEnd*(*s*)

if (*new\_list.size*() == *N*)

cmts = new CMTS (*new\_list*)

*new\_list.removeFirst*()

else cmts = null

Return *new\_list* and cmts

The order of this algorithm is O(1). The measurement needs N valid consecutive measurements to form a cmts that is used to find a set of possible tasks states. When at least one of the measurements out of the N is incorrect the algorithm cannot find any possible task state.

## 5 SIMULATIONS

### 5.1 The ANTS challenge problem

The ANTS problem uses an array of Doppler sensors to detect and track targets located in its surrounding area. Every Doppler sensor is divided into three separate sectors, each responsible for a 120 degree segment. At any given time only one sector may be active. Each sensor has only partial information about

sensed targets. The Doppler may have information about the distance of the target as a function of the angle, while the angle figure is unknown, and about the radial velocity, which is the velocity of the target towards the sensor. Note that having an arc that a target is located on and its radial velocity is not equivalent to having the location and velocity of the target.

In the ANTS challenge problem Doppler sensors may have associated noise when sensing targets. This noise leads to misleading arc data and a wrong radial velocity. We were interested in finding a good way to overcome noisy data and produce reliable results in real time.

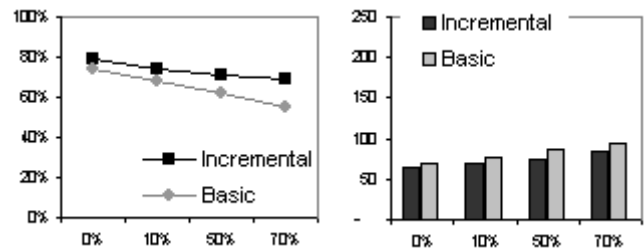


Figure 1: Sampling noise. Incremental vs. basic DDM – target tracking percentage and average time.

We simulated a geographic area of 1200 by 900 meters in our experiments. 30 targets were passed into the supervised area with a random initial location, direction and velocity (of up to 50 km. per hour). Each target, upon leaving the area, resulted in a new target entering the area. The new target arrived from the exit location and had the same velocity. A random velocity and direction implies that target remains in the area for a random time. For instance, we saw that 29% of the targets stayed less than 60 seconds in the controlled zone. In the basic setting there were 20 Dopplers in the environment. Each Doppler had a random location, direction and velocity of up to 50 kilometers per hour. Each sector had a maximum range of 200 meters. Each experiment simulated one hour. The criteria for fitness of a target to a calculated path was: (1) the distance between the calculated Location(t) and the real Location(t) was less than 1 meter, and (2) the difference between the calculated V(t) vector and the real V(t) vector was less than 0.1 meter per second. We considered two kinds of noise in our simulations: sampling noise and communication noise.

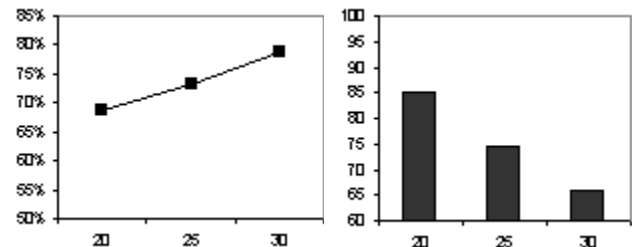


Figure 2: 70% sampling noise. Target tracking percentage and average time as a function of the Doppler number.

## 5.2 Sampling noise

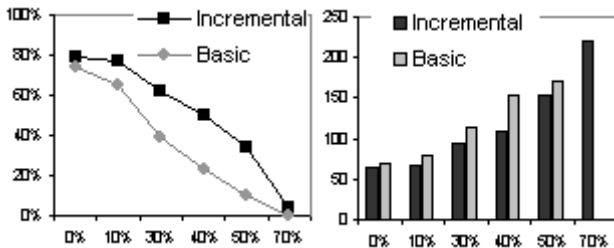
Using sampling devices leads often to inaccurate measurements. Many sampling systems strive to balance quantity and quality. We investigated the consequences of using small numbers of accurate sensors versus large numbers of inaccurate sensors. One of the key questions is how tolerant the system is to the inaccuracy of its sensors. Sampling noise tends to be related to the distance from the center of the sensor to the sampled target. We applied the following sampling noise model:

$$1. F(r) = x \cdot r^2 / R^2$$

where  $x$  is the percentage of noise. We refer to  $x$  as the noise factor,  $r$  is the distance from the center of the sensor to the sensed target and  $R$  is the maximum detection range. We wanted to study the effect of  $x$  on the performance of IDDM.

Figure 1 presents the performance of the IDDM model versus the basic DDM model when the sensors suffer from sampling noise. The decrease in the percentage of the tracked target was more moderate in the IDDM. Moreover, the IDDM could track the targets about 10% faster regardless of the noise factor. The results indicate that when using faulty and noisy sensors IDDM was preferred over the basic DDM model.

We used sensors with 70% sampling noise to study the influence of the sensors quantity on the target tracking percentage and the tracking average time. Figure 2 presents the performance of IDDM when using 20, 25 and 30 such sensors. As the number of Dopplers increased the average tracking time decreased while the tracking percentage increased. Comparing Figure 1 and Figure 2 shows that using 30 inaccurate Dopplers with 70% noise factor is equivalent to using 20 accurate Dopplers. Using 25 Dopplers with a sampling noise of 70% results in the same tracking percentage as when using 20 Dopplers with 10% sampling noise. The results indicate that the IDDM is very tolerant towards sampling noise as it allows compensation in terms of participating sensors.



**Figure 3: Communication noise. Incremental vs. basic DDM – target tracking percentage and average time.**

## 5.2 Communication noise

In a real-time communication system we often face the decision of whether to invest resources to improve the reliability of the communication channel. Investigating this dilemma is one of the goals in a fault tolerant communication system. In the case of the ANTS domain, there are communication channels leading from the Dopplers up the hierarchy. We applied a

communication noise model. In this model there was a certain probability that a message sent by a Doppler did not reach the cl due to faulty or noisy communication channels. We applied a constant noise probability to imitate communication noise. Such kind of noise represents noise in many systems that use communication ([13]).

$$2. F = x$$

Examining Figure 3 one can see that both the DDM and the IDDM can manage with sampling noise of 10%. However, while the noise factor influenced dramatically the performance of the DDM, starting with very small factor, the number of tracked targets when using the IDDM decreased more slowly. The DDM could track only 23% of the targets when faced with a noise factor of 40% while the IDDM tracked 50% of them. With a noise factor of 50%, the number of the tracked target dropped to 10% when using the DDM while it was 34% when the IDDM was used. Looking at the tracking average time one can see that the IDDM improved the times by about 10%. However, only the IDDM could cope with a 70% noise factor.

## 6. RELATED WORK

The benefits of hierarchical organizations have been argued by many. So and Durfee draw on contingency theory to examine a variety of benefits of hierarchical organizations; they portray a hierarchically organized network monitoring system for task decomposition and also consider organizational self-design [8]. DDM differs in its organization use to dynamically balance computational load and in its way to support mobile agents.

The idea of combining partial local solutions into a more complete global solution goes back to early work on the distributed vehicle monitoring testbed (DVMT) [14]. DVMT also operated in a domain of distributed sensors that tracked objects. However, the algorithms for support of mobile sensors and for the actual specifics of the Doppler sensors themselves is novel to the DDM system. Within the DVMT, Corkill and Lesser investigated various team organizations in terms of *interest areas* which partitioned problem solving nodes according to roles and communication, but were not initially hierarchically organized. Wagner and Lesser examined the role that knowledge of organizational structure can play in control decisions [10].

The DDM is discussed thoroughly in ([11]). It took advantage of the hierarchical organization and used it as an efficient way of reducing the uncertainty associated with solutions based strictly on local information by combining partial local solutions into a global solution, and as a structure for building more fault tolerant systems. The incremental DDM takes the advantages of the DDM one step further and increases dramatically its fault tolerant capabilities: it detects misleading information and makes use of a recycling mechanism to allow reuse of accurate information.

Alternative approaches to realtime distributed resource allocation are being explored within the ANTS program [4,15]. All of those methods assume that agents are stationary. Vincent *et al* use a limited team organization, assigning agents to sector managers [16]. Each manager is fusing information for

tracking; a hierarchical organization serves to limit communication among agents and to provide a measure of fault tolerance: if a sector manager is disabled, another can fill in. Approaches within the ANTS program that deal with noise involve filtering processes combined with heuristics that set a threshold on measurements: the lower the measurement, the more likely the distance to the target is large. If the measurement is less than the noise threshold, the measurement is discarded as unreliable.

In the simulations used in the other ANTS projects, sensors are stationary and must coordinate their measurements with other sensors to obtain accurate estimates of a target. The way the information gathered by a sensor is used to find the exact location and velocity of a sensed target is to intersect three arcs taken at the same time by different sensors. Each of the three arcs represents a possible location of the target. The results of this method are accurate and fast if the data is not noisy. However, the need for coordination among three mobile sensors in order to take measurements of the same target at exactly the same time is hard to fulfill. Any lack of synchronization leads to inaccurate results. Moreover, having noisy sensors, the agreement among the three sensors may result in wrong locations and velocities of the sensed targets. It takes only one noisy sensor out of the three to lead to a misleading result as the agreement takes into consideration the three sensor data.

Ammar *et al* ([12]), discuss issues that arise in hardware and software fault tolerant systems; they also analyze the impact of fault tolerance requirements on system design and the impact of fault tolerance provisions on system reliability. Their work suggests redundancy as a basis for fault tolerance. They categorize three kinds of redundancies: temporal redundancy, spatial redundancy, and informational redundancy. In our work we use redundancy of computations as we compute more capsules than the DDM; therefore IDDM has temporal redundancy. As we have suggested, to overcome noise one should use more sensors; therefore IDDM also has spatial redundancy. Sampling the same targets from different points of view leads to data redundancy; therefore IDDM also has informational redundancy. Having these redundancies is obtained by applying a large-scale organization of unsophisticated agents as suggested by the IDDM model.

## 7. SUMMARY

In this paper we have introduced a hierarchical approach to the realtime management of large-scale task and team environments where agents must identify the changing states of task-entities. We have developed several algorithms for each type of node in the hierarchy. We have extended the DDM system, described elsewhere, with incremental algorithms to handle noisy environments. Through extensive experiments in a simulated domain, we demonstrated the ability of IDDM to process noisy local information to produce partial solutions to local tasks. The system can then integrate local information from sensors into a more accurate global solution, and it does so in real time.

In order to apply IDDM in a different environment, it is only necessary to define two domain specific functions with low complexity: *PosTS* that maps measurements to

possible states and *ResBy* that determines whether one given task state can be the consequence of another given task state. Give these functions, all the algorithms of the IDDM that were implemented for the ANTS domain can be applied. Thus, we believe that the results obtained for the ANTS simulations are general and will be obtained in any such domain.

## 8. REFERENCES

- [1] Pöss, Christian Doppler in Banska Stiavnica, in *The Phenomenon of Doppler* (Prague, 1992), 55-62.
- [2] Kerut, Edmund Kenneth, Handbook of echo-doppler interpretation, Amrmonk, N.Y.: Futura Pub. Co., 1996.
- [3] Gill, Thomas P., The Doppler effect: an introduction to the theory of the effect, London: Logos, 1965.
- [4] Leen-Kiat Soh and Costas Tsatsoulis. Reflective Negotiation Agents for Real-Time Multisensor Target Tracking, IJCAI-01, Vol. 2, 2001.
- [5] Richard J. Doviak and Dusan S. Zrnica. Doppler radar and weather observations. Orlando, Academic Press, 1984.
- [6] ANTS Program Design Document, unpublished.
- [7] So, Y., and Durfee, E.H., "Designing Tree-Structured Organizations for Computational Agents," *Computational and Mathematical Organization Theory*, 2(3), 1996.
- [8] Young-pa So and Edmund Durfee, "A distributed problem-solving infrastructure for computer network management, International Journal of Intelligent and Cooperative Information Systems, 1992.
- [9] D. Corkill and V. Lesser, "The use of meta-level control for coordination in a distributed problem solving network," IJCAI 1983.
- [10] T. Wagner and V. Lesser, "Relating Quantified Motivations for Organizationally Situated Agents," *ATAL 2000*.
- [11] Osher Yadgar, Sarit Kraus and Charles L. Ortiz, Jr., Hierarchical organizations for real-time large-scale task and team environments, unpublished.
- [12] H. H. Ammar, B. Cukic, C. Fuhrman, and A. Mili, A Comparative Analysis of Hardware and Software Fault Tolerance: Impact on Software Reliability Engineering, *The Annals of Software Engineering*, Vol. 10 (2000), Fall 2000.
- [13] E. Kushilevitz and Y. Mansour, Computation in noisy radio networks, in Proc. 9th Ann. ACM-SIAM Symp. on Discrete Algorithms, 1998, pp. 236-243.
- [14] Lesser, V.R., Corkhill, D.D., and Durfee, E.H. *An update on the DVMT*, CS Technical Report 87-111, University of Massachusetts, Amherst, 1987.
- [15] Proceedings of the AAAI Fall Symposium on Negotiation Methods for Cooperative Systems, AAAI Press, 2001.
- [16] Vincent, R., Horling, B., Lesser, V. and Wagner, T. "Implementing Soft Real-Time Agent Control." In *Autonomous Agent 2001*, 2001, AAAI.