Goal-satisfaction in large-scale agent-systems: a transportation example ⋆

Onn Shehory
The Robotics Institute
Carnegie-Mellon U.
5000 Forbes Ave
Pittsburgh, PA 15213, USA
onn@ri.cmu.edu
Tel: +1-412-268-3740
Fax: +1-412-268-5569

Sarit Kraus
Dept. of Math and CS,
Bar Ilan University,
Ramat Gan, 52900 Israel
sarit@cs.biu.ac.il
and
Inst. Adv. Comp. Studies,
Maryland U., College Park, MD USA

Osher Yadgar
Dept. of Math and CS,
Bar Ilan University,
Ramat Gan, 52900 Israel
yadgar@cs.biu.ac.il
Tel: +972-3-5318863
Fax: +972-3-5353325

Abstract. A framework for cooperative goal-satisfaction in large-scale Multi-Agent Systems (MAS) is presented in this paper. This is performed by demonstrating the applicability of a low complexity physics-oriented approach to a large-scale transportation problem. The framework is based on modeling cooperative MAS by a physics-oriented model. According to the model, agent-systems inherit physical properties, and therefore the evolution of the computational systems is similar to the evolution of physical systems. We provide a detailed algorithm to be used by a single agent and implement this algorithm in our simulations. Via these we demonstrate effective task allocation and execution in an open, dynamic MAS that consists of thousands of agents and tasks.

1 Introduction

Goal-satisfaction in MAS may require cooperation among the agents, but cooperative goal-satisfaction may be beneficial even if the agents can perform goals by themselves. Traditional task-allocation methods [14] require coordination via communication [3]. In very large agent-communities there usually cannot be direct, on-line connection between all of the agents, as such a connection is too costly. Therefore, when the number of agents increases, the complexity of most of the cooperation methods becomes unbearable. To resolve the scale-up computational explosion of cooperation mechanisms in large MAS we present a different approach.

We apply a model based on methods from classical mechanics [12] to model large-scale agent-systems. The physics-oriented methods are used to construct a beneficial cooperative goal-satisfaction algorithm to be used by the single agent within the system. In spite of the myriad differences between particles and computational agents, we show via simulations that, at least for the example problem that we have tested, using the physics-oriented approach enables effective cooperation and goal-satisfaction in very

⋆ This material is based upon work supported in part by the NSF under grant No. IRI-9423967, ARPA/Rome Labs contract F30602-93-C-0241 (ARPA Order No. A716) and the Army Research Lab under contract No. DAAL0197K0135.

large agent-systems. In current research we are investigating the applicability of our model to other, non-physical domains.

Many problems arise in large scale MAS research. In this paper we concentrate on investigating one facet – task allocation and execution within large-scale cooperative MAS[2]. More specifically, we consider cases in which cooperative autonomous agents allocate themselves to tasks. We describe a model that allows for the dynamic agent-task allocation and is appropriate for large-scale MAS and test it. The latter is performed by simulating a dynamic agent system that follows our suggested mechanisms and consists of thousands of agents and tasks. To our best knowledge, up to date, this is the largest simulation of a task allocation and execution in a dynamic, open MAS. The model we present provides a solution to problems which were not addressed previously in MAS, and may be the basis for future solutions for a larger class of problem domains. We show here applicability to one domain and in research in progress we have shown applicability to another, less physical problem domain. Yet, more research is necessary to determine applicability to additional domains.

1.1 Assumptions, notations and concepts

We assume that the agents with which we deal have the ability to perceive the virtual[3] displacement in the goal-space, and can perceive the properties of other adjacent agents and goals. This may be done by sensors integrated into the agents. We also assume that each agent knows about the types of resources that other agents may have, but may be uncertain as to the particular resource-holdings of any other individual. These two assumptions are necessary since the agents are expected to propagate from state to state within the goal-space according to the properties of the surrounding agents, goals and obstacles. In order to enable such propagation, some knowledge regarding neighbors is necessary. We assume that each agent has a performance capability that can be measured using standard measurement units. The standard measurement will be used as a quantitative way of measuring the agents' success in fulfilling goals. In addition, we assume that there is a scaling method which is used to represent the displacements of the agents in the goal-space and to evaluate the mutual distances between goals and agents within this space. This assumption is necessary since virtual distances (or physical distances) are a significant factor in the model we present. We assume that goal-satisfaction can be achieved progressively. That is, a goal may be partially satisfied at one instant, and its remaining non-satisfied part may be complete at another point in time.

To present our model, we review concepts and notations from physics. The displacement vector of a particle i is denoted by $\mathbf{r}_i$. $\mathbf{v}_i$ denotes the velocity, and $\mathbf{a}_i$ denotes the acceleration. The kinetic energy of a particle i is represented by k_i, and the potential is represented by V. The potential is a spatial function and therefore is sometimes called a

[2] Cooperative MAS are frequently referred to as a Distributed Problem Solvers (DPS) [2] agent systems. In DPS agent systems as in cooperative MAS, agents attempt to increase the common outcome of the system.

[3] Since the goal-space is not necessarily physical, we do not assume physical distances and therefore call them virtual. In work in progress we show how such virtual distances can be modeled and computed.

field of potential or a potential-well. Forces can be derived from the potential. Each particle i's mass is denoted by m_i, its displacement is denoted by the displacement vector $\mathbf{r}_i$, its momentum by $\mathbf{p}_i$ and the force that acts on it is denoted by $\mathbf{F}_i$.

1.2 Adapting physics to MAS

MAS	Physics
identifying the environments where physics-oriented models are appropriate; matching particle properties to agents/goals	locating particle models and their properties
selecting the matter-states that can be used to model automated-agents' systems.	identifying states of matter and the particle behavior within
developing algorithms for agents' goal-satisfaction; adjusting to the physics system for validity of the algorithm	using mathematical formulation to predict and describe the properties and evolution of the selected particle model
analysis of the complexity and properties of the algorithm	theoretical and simulation-based analysis of physical particle systems behavior

Table 1. Distributed AI and Physics for cooperative MAS

In the MAS that we consider, there is a large set of agents and a large set of goals they need to satisfy. Each agent has capabilities and should move toward satisfying goals. We use a physics model that consists of particles which represent the agents and the goals, and to develop a distributed cooperative goal satisfaction mechanism. We first step match between particles and their properties, agents and their capabilities, and goals and their properties (see table 1). Next, we identify the state of matter for modeling a community of agents and goals. The mathematical formulation that is used by physicists either to describe or to predict the properties and evolution of particles in these states of matter, serve as the basis for the development of algorithms for the agents. However, several modifications of the physics model are necessary to provide an efficient algorithm for automated agents.

In our model, agents and goals are modeled by dynamic particles and static particles, respectively. The match between particle properties and agent/goal properties is described in table 2. We model goal-satisfaction by a collision of dynamic particles with static particles. However, the properties of particle-collisions are different from the properties of goal-satisfaction and several adjustments are needed in order to provide the agents with efficient algorithms. These modifications are described in detail in this paper.

Automated Agents	Physics Model
community of agents satisfying goals	non-ionic liquid system
agent	dynamic particle
goal	static particle
agent's capabilities	particle's mass
agent's (virtual) location in agents-goals space	location of particle
goal satisfaction	static-dynamic collision
algorithm for goals allocation	formal method for calculating the evolution of displacement

Table 2. The match between the physics model components and the large-scale automated agents environments

2 Modeling agents – a physics-oriented approach

Classical mechanics provides a formal method for calculating the evolution of the displacement and the momentum of classical particles. For a particle i, the equations of motion are:

$$\mathbf{F}_i = m_i \ddot{\mathbf{r}}_i = m_i \mathbf{a}_i \quad \text{and} \quad \mathbf{p}_i = m_i \dot{\mathbf{r}}_i = m_i \mathbf{v}_i \tag{1}$$

The motion of a particle depends on the field of potential in which it moves and the force $\mathbf{F_i} = -m_i \boldsymbol{\nabla}_{r_i} V(\mathbf{r})$. The model we present entails treating agents, goals and obstacles as particles. That is, each agent will have its equations of motion and an initial state. Note that an agent's equations of motion do not necessarily entail real physical motion. The potential field in which an agent acts represents the goals and the other agents in the environment. Subject to the potential field, agents solve the equations of motion and, according to the results, progress towards the solution of goals and either cooperate or avoid conflicts with other agents. The cooperation and conflict-avoidance are emergent properties of our physics-oriented model.

An appropriate physical system must consist of a potential that, when adapted to the agent-model, will lead the agents to successful and beneficial goal-satisfaction. The fluid model is most appropriate for our systems. As opposed to the solid state, a fluid system can evolve from its initial state into new, different states. Preferable is a model that does not require long range interactions (e.g., the non-ionic liquid model). In the model suggested in [12] the typical potential of a particle i in a non-ionic liquid was suggested (the Lennard-Jones potential). In the model developed for the specific transportation application dealt with in this paper we experimented with several different potential functions and finally concentrated on the following:

$$V(\mathbf{r})_{ij} = \gamma(\alpha \ln r_{ij} + \beta r_{ij}^{-2} + \chi r_{ij}^{-4}) \tag{2}$$

where r_{ij} corresponds to the distance of particle i from particle j. This potential diminishes after a short distance, thus implying that the interaction between the particles in the system is limited to short distances.

3 The physics-agent-system (PAS) model

The cooperative MAS system with which we deal is modeled by a set of particles and a potential field. The agents in the system are modeled by dynamic particles and their potential-wells. The goals and the obstacles are modeled by static particles which are represented by fixed potential-wells. The superposition of the potential-wells of the particles, either agents or goals and obstacles, constructs a potential field. The particles move according to the field of potential and their own properties.

In the PAS model, the agent's capability of satisfying goals is represented by the mass of the particle that models it, and therefore by the potential-energy $k = mv^2/2$, which is a product of the mass, as well. Particles with a greater potential-energy model agents that can satisfy larger or more difficult goals and sub-goals. This means that a greater mass of a dynamic particle that models an agent (other properties remaining constant, and thus causing a greater potential-energy), entails a larger capability of goal-satisfaction by the agent. The mass of a fixed particle represents the size of the goal or the obstacle. This means that in order to satisfy a greater goal, which is modeled by a particle with a greater mass, more efforts are necessary on the part of the agents.

The displacement vector of a particle $\mathbf{r_i}$ models the displacement of the agent in the goal-space. According to the virtual displacement of an agent, its distances from other agents, goals and obstacles can be calculated. The potential is calculated according to these distances. The momentum vector $\mathbf{p_i}$ of particle i represents its physical velocity and is used for the calculation of the kinetic energy. In the PAS model, the velocity of a dynamic particle represents the rate of movement towards the satisfaction of a goal or a part of a goal.

3.1 Motion towards goal-satisfaction

In the physical world, the motion of particles is caused by the mutual attraction between them. In the agents' system, the agents calculate the attraction and move according to the results of these calculations. The reaction of a particle to the field of potential will yield a change in its coordinates and energies. In our model, each agent will calculate the effect of the potential field on itself by solving a set of differential equations. According to the results of these calculations, it will move to a new state in the goal-domain (section 3.3).

The steep decay of the potential function beyond a short distance from the center of the potential-well results in derived weak forces and negligible interaction. Physicists have shown that when the long-distance interactions are neglected, the results of simulations still agree with theoretical statistical-mechanics and thermodynamics [15, 11]. Therefore, it is common to cut off the range of interaction by cutting off the potential function after it diminishes to from 1 to 10% of its maximal value. The radius of interaction (and of the cut-off) is denoted by r_I.

Agents will use numerical integration to solve the equations of motion that they must solve, with respect to time. The integration must be iterated frequently and performed with small time-steps dt. We determine the size of the time differential dt relying on the experience gathered in physics simulations [11]: we demand that a typical particle in the model will pass a distance of r_0 in ~ 10 time-steps dt. This requirement implies that the average velocity $\overline{v}$ of a particle (at its initial displacement) directly affects dt by the relation $dt = r_0/\overline{v}$.

3.2 Collision and goal-satisfaction

The dynamics of the physical system which models the computational system leads to collisions between particles. Two types of collisions are possible: a collision between two dynamic particles, which we denote by DDC, and a collision between dynamic and static particles, denoted by SDC. In our model, the DDC represents the interaction between two agents. In order to prevent situations where agents overlap, the particles that model the agents have a mutual repulsion. The decision on which agents shall perform a specific goal will emerge from the repulsion. Dynamic particles that model agents shall have a potential that consists of a dominant repulsive component.

The SDC represents agent-goal interaction. In such interactions we would like the static particle that models the goal to attract the dynamic particle that models the agent. Adopting physical concepts, we use the notion of typical radius to specify the point from which the particle starts the collision. A typical radius σ of a particle is usually taken to be the distance from its center to the point wherein the force is zero. An SDC occurs when a dynamic particle is in the vicinity of a static particle. Vicinity here means that the distance between them is a few typical radii (r_0).

The goal-satisfaction is performed during the collision. An agent that reaches a goal may either completely or partially satisfy it. In both cases, the model requires a reduction in the magnitude of the goal. This implies that the mass of the modeling particle shall be reduced, but mass-reduction is not a physical property of such a collision. Therefore, some modifications of the model shall be done, as long as they do not affect the general evolution of the system. This will be possible if the model consists of a scheme for a temporal partition of the evolution of the system. This means that the evolution of the system will be partitioned into several time segments (different from dt, much longer), and in each temporal segment the physical evolution of the system will not depend on the other segments.

3.3 A protocol for the single agent

In order to cause evolution of the system towards goal-satisfaction, each agent uses the information that it can gather by observation (e.g., via sensors) about its neighboring agents and goals and regarding its previous state. According to this information, the agent will construct the local field of potential and solve the equations of motion. The results of the equations of motion will enable the agent to decide what its next step towards goal-satisfaction will be. The exact detailed algorithm for the single agent i is as follows:

Loop and perform the goal-reaching and goal-satisfaction processes until the resources necessary for satisfying goals have been depleted or no goals within the interaction range r_I have been observed for several time-segments.

Goal-reaching process

1. Advance the time counter t by dt.

2. Locate all of the agents and goals within the range r_I, the predefined interaction distance. Denote the distance to any neighboring entity j by r_{ij}.
3. Calculate the mutual potential (using equation 2) with respect to each of the agents and goals within the range.
4. Sum over all of the pairwise potentials $V(r_{ij})$ and calculate the gradient of the sum to derive the force F_i.
5. Using F_i and the previous state $\mathbf{r_i}(t-dt), \mathbf{p_i}(t-dt)$, solve the equations of motion as described in section 2, in equation 1.
6. The results of the equations of motion will be a new pair $\mathbf{r_i}(t), \mathbf{p_i}(t)$. Move to the new state that corresponds to the displacement $\mathbf{r_i}(t)$.
7. At each time-step, after moving to a new state, calculate the new kinetic energy and potential according to the new coordinates $\mathbf{r_i}(t), \mathbf{p_i}(t)$.
8. If your distance from the center of a particle that models a goal is greater than r_0, return to step 1. Otherwise, start the goal-satisfaction process.

The goal-satisfaction process
After reaching a goal, the agent must satisfy all or at least parts of it:

– Move into the potential-well that models the goal according to the physical properties of the entities involved in the process and perform the goal.
– If m_a, the mass of the particle that models the agent, is smaller than m_g, the mass of the particle that models the goal, subtract m_a from m_g. Else, $m_g = 0$. In a case of depleting resources, m_a is reduced in a similar way. Return to step 1.

The iterative method which we propose leads to a gradual reduction in the amount and size of the goals to be satisfied, and will lead finally, to completion of the goals.

4 Simulation

To examine our model and show its applicability to real problems we have performed a set of simulations. Via these we demonstrate effective task allocation and execution in an open, dynamic MAS that consists of thousands of agents and tasks. The problem domain for which the simulations where performed is as follows. We simulate freight deliveries within a metropolitan. Such problems in real environments are commonly solved by having one or a few dispatch centers to which delivery requests are addressed and these each centrally plans and accordingly allocates delivery tasks to delivering agents. This method may face bottlenecks and inefficiency when a large number of agents and tasks is present. We demonstrate how the PAS model can overcome this limitation.

We consider the road-network of a large metropolitan. A snapshot of a part of this network is depicted in figure 1. In this figure squares represent messengers and circles represent tasks. The city map is represented by a lattice-like graph. The boundaries of the city are $20,000 \times 30,000$ meters. The lattice includes vertices located 200 meters apart from each other. An edge may exist between each two neighboring vertices. Each vertex represents a junction and each edge represents a road between two junctions. We designate the map ”Full Lattice” when each vertex has edges enamating to all of its neighboring vertices. A more realistic map would have some of the edges missing. To obtain such a map we use some probability to determine the existence of each edge. As a result disconnected sub-graphs (designated clusters) may occur. In such cases the largest cluster will be selected to represent the city. We designate the map ”X% Lattice” when lattice and cluster generation are performed taking the probability of including an edge in the lattice to X%. Note that the structure of cities and roadways regulations may prevent movement along the shortest path between two locations, as assumed by the general algorithm. Thus, in the simulation, the distance between two locations l_1 and l_2 was calculated as the shortest way that one could drive from l_1 to l_2. Furthermore, if the direction for movement[4] $\hat{v}$ calculated by the agent in the goal-reaching process algorithm does not agree with a road direction $\widehat{road}$, then the road with the smallest angel with $\hat{v}$ is selected for movement. This selection is not different from a physical behavior in environments with obstacles, and therefore justified.

The simulation consists of iterations in which new freights dynamically appear at random locations on the map. The freights have an initial size which is set to 1 kg in the homogeneous case and to a random value (out of a given range) in the heterogeneous cases. In addition, each freight has a random destination. Messengers (agents) follow our algorithm to perform tasks of reaching freights and delivering them to their destination.

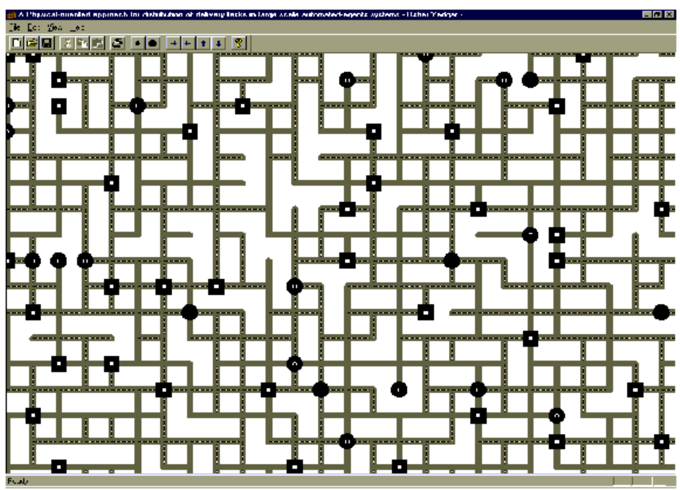

Fig. 1. A fragment of city map

We have performed several different types of simulations. These varied over the amount of tasks and agents involved, the homogeneity of agents and tasks, the reliability of communication and the intensity of the lattice map.

Our simulations were initially performed such that agents and tasks are homogeneous in the sense that they have similar capabilities and capacities. We started with

[4] The notation $\hat{v}$ refers to the direction of a vector $\mathbf{v}$.

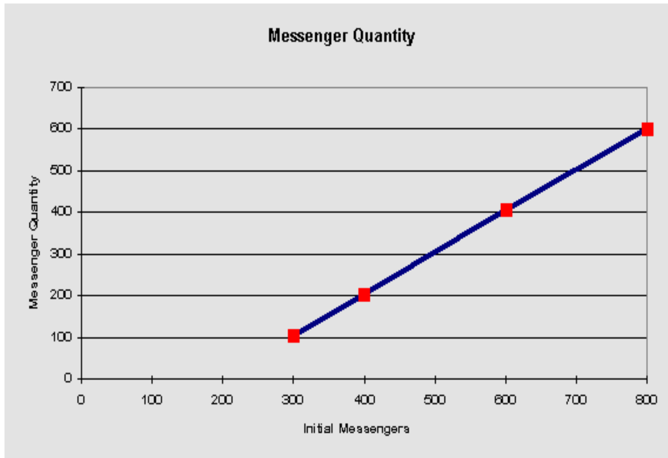

Fig. 2.

these since they are simpler to handle and predict. However it was necessary to examine cases in which agents and tasks are not homogeneous, which are more realistic. In the homogeneous case, masses of particle were set to $1kg$, whereas in the heterogeneous case masses where set randomly out of a given distribution. We have also examined several lattice maps, starting from a full lattice and moving to 90% and 80% lattice maps. Since we have seen no significant difference in the performance between the different maps, we concentrated on the 90% lattice map. To learn the effect of unreliable communication on the performance we have experimented a case in which messages are passed with arrival probability which is smaller than 1. Additional parameters of the simulations are as follows. During the simulation no new messengers appear. Parameter values are $\gamma = 1$ $\alpha = 4000$, $\beta = -15E5$, $\chi = 5E11$ (these are used in equation 2), R_0 is 100 meters, R_I is 2,000 meters. Note that these values where not arbitrarily chosen. Rather, we have experimented with a variety of values to fine-tune the system until we arrived at these coefficients. We sought timely task performance, and these coefficients yielded the best results.

In the homogeneous case, we considered five settings of agent and task quantities. In the 4 simulation settings in which the number of agents was 300, 400, 600 and 800 the initial number of tasks was 1200. In the case of 1200 agent the initial number of tasks was 1500. In all 5 settings additional tasks where arriving at a rate of 600 tasks per hour. The different quantities of agents in the first four settings allowed us to study the effect of the number of messengers (hence the messengers/freights ratio as well) on the system's performance. The fifth setting was aimed mainly at studying the effects of up-scaling.

The main results of the simulations are summarized in the graphs below.

– In figure 2 the ratio between the number of messengers in the system and the number of agents that are simultaneously involved in movement towards tasks is presented. The term *Messenger quantity* is the number of messengers which are currently moving towards freights. The other messengers are performing tasks. From the graph one can observe that as the number of messengers involved increases, so does linearly increases the number of those that simultaneously move towards tasks. This result for itself does not seem of merit, however it results in reduction in the time required for task execution (as can be seen in figure 4).

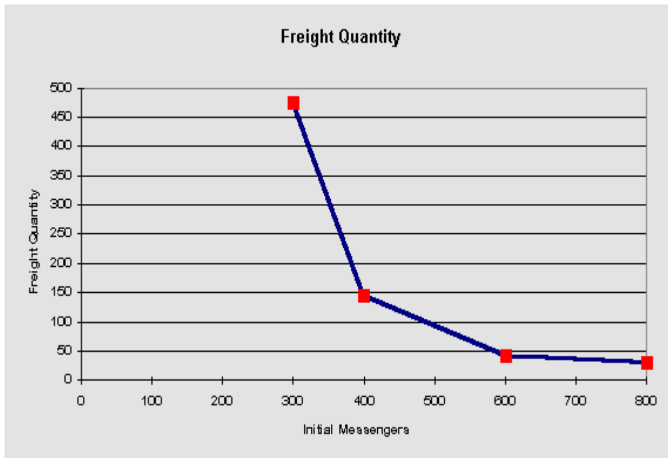

Fig. 3.

– The term *Freight quantity* in figure 3 is the number of freights currently waiting for a messenger to deliver them. We observe that this number drops sharply as the quantity of messengers goes up. The critical point where transition occurs is around 500 messengers. Given that 1200 tasks are present, this means that for significantly lowering the number of freights which are simultaneously waiting to be delivered it is enough to have a ratio of around 0.4 between messengers' and tasks' quantities in the system. Increasing the ratio over 0.5 does not bring about a significant increase in the performance (with respect to the numbers of freights waiting to be delivered).
– The term *Fulfilling messenger reaching time* in figure 4 refers to the time[5] it takes a messenger, who successfully delivers a freight to its destination, to reach this freight. One can observe that as the quantity of messengers increases (and so does their density), the time which is required for a messenger to reach a freight increases as well. This is a disadvantageous property, however it does not mean that increasing the density is all bad. As we have seen before - it significantly reduces the number of freights which simultaneously wait for being delivered. In addition, as shown in figure 5, the average waiting time of the freights decreases as well.
– In figure 5 the freight average waiting time is presented. The term *Fulfilled freight waiting time* refers to the time that a freight that was successfully delivered to its destination has been waiting before being handled by a messenger. A sharp re-

[5] Here and in the following graphs time is measured in seconds.

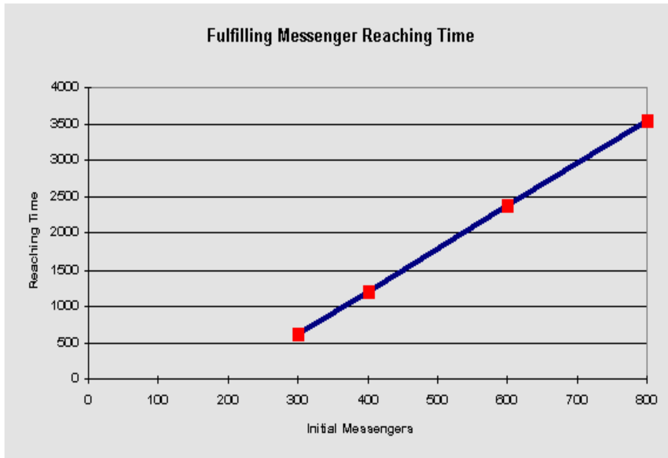

Fig. 4.

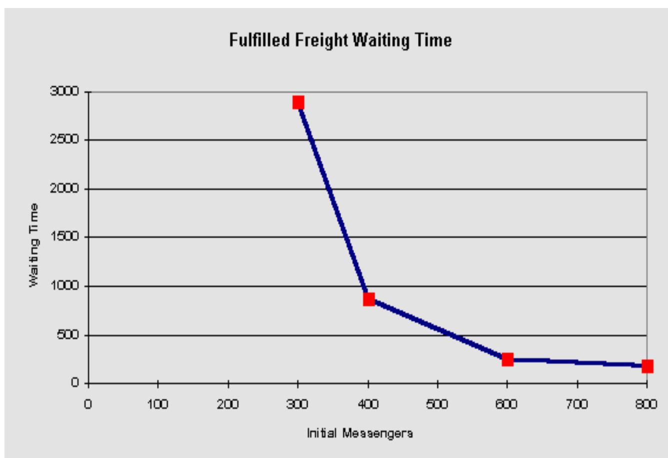

Fig. 5.

duction in the waiting time is observed. We observe phase transition around 500 messengers, similar to the phase transition in the case of *Freight quantity* (figure 3). This further supports the observation that it is not worth while to increase the agent/task ratio to above some ratio which is, in our simulation settings, around 0.4 to 0.5.

– Figure 6 presents the average *Freight fulfillment time* which is the time between the freight initiation and its arrival at its destination. Less steep than in previous graphs, yet clear, is the improvement in the performance reached around 500 messengers. It is important to notice that for 600 messengers and more the task execution time is less then 1500 seconds. For a city of the size with which we deal (20×30 km)

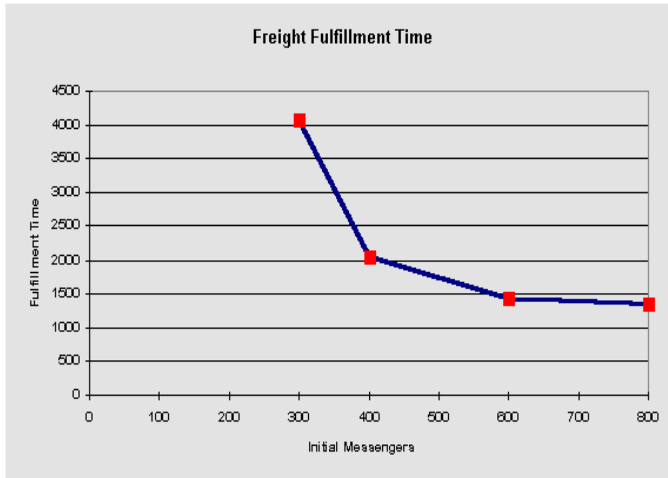

Fig. 6.

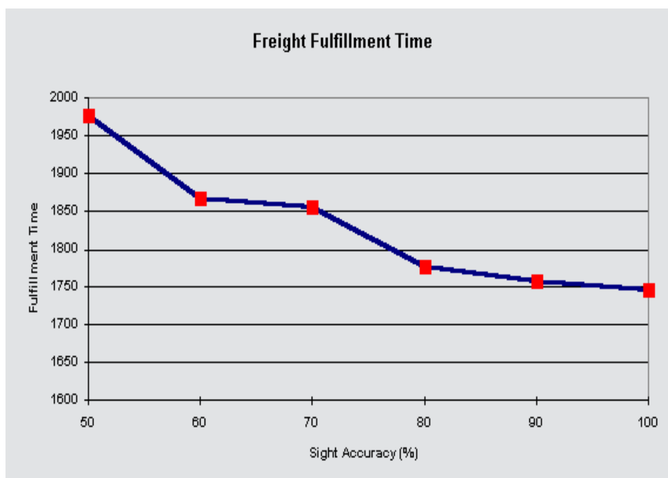

Fig. 7.

with a speed limit of 50km/hr, this is a desirable fulfillment time.

– Figure 7 presents one of the results of a set of simulations of heterogeneous ensembles of agents and tasks, where the probability of message reception varied between 50% and 100%. That is, in this simulations an agent may not receive some of the information regarding neighboring tasks and agents although this was transmitted. The initial masses of tasks was set randomly between $1kg$ and $100kg$, while the masses of the agents was set randomly between $80kg$ and $180kg$. If the capacity of an agent was smaller than the size of the task, it delivers only part of the task at a time. The number of agents in this set of simulations was 600 and the initial num-

ber of tasks was 1200. The other parameters were as in the previous simulations reported above.
Our results indicate that the heterogeneity of the agents does not significantly change the behavior of the system. From figure 7 we can conclude that the "Freight Fulfillment Time" increases linearly when the probability of messages arrival decreases. However, even with 50% arrival of messages, the fulfillment time is better than in the case of 400 messengers with 100% arrival of messages (see figure 5). Similar results were obtained with respect to the other parameters.

From the results presented above as well as myriad additional experiments (which weren't presented here for space reasons) we conclude the following:

- The PAS model can be applied for use in large scale agent systems to solve real problems.
- An increase in the number of agents in the system does not increase the amount of computations per agent. Thus, larger systems do not require more computation time.
- An increase in the number of agents in the system, holding the number of tasks constant, is beneficial only to some extent. Beyond some agents/tasks ratio, no significant improvement in performance is observed. We believe this phenomenon results from redundancy in densely populated agent systems.
- The results observed are similar for different densities of the lattice map used as well as for low probabilities of unreliable communication channels. They become better when the the distribution of tasks is not even, as typically happens in the center of large metropolitans.

5 Related work

The issue of allocating agents to goals has widely been discussed among DAI researchers. The Contract Net Protocol [14] uses negotiation based on task announcements, bids and contracts for task allocation. While the CNP is based on the exchange of information, the model we present minimizes the transmitted information and thus enables large-scale systems to be efficient. A study of planning in large-scale agent-systems has been presented in [17, 16]. In that research, the general-equilibrium approach from economics serves as the theoretical basis for the planning mechanism. We also discuss large-scale systems and apply an analytical model for designing the distributed planning mechanism, however we use a physics-oriented approach for cooperative MAS, not for competitive agents.

A large body of DAI research studies coordination among agents for distributed problem solving (for example, [2], PGP [5], GPGP [1], [6], [18]). In [4], Durfee and Lesser study the Partial Global Planning (PGP) approach to coordination by implementing it in the Distributed Vehicle Monitoring Testbed (DVMT). The DVMT is a network of vehicle monitoring nodes. Each node has a planner that plans incrementally. Nodes do not communicate their detailed actions, but do communicate according to a meta-level organization. A PGPlanner modifies local plans as required due to incoming messages. In its incremental planning and restricted communication the PGP model is similar to our model. The DVMT task domain which was used as a testbed for both PGP and GPGP includes monitoring traffic and directing it. This is performed by the agents generating tentative maps for vehicle movements in their areas. Our transportation framework is different: we require that a transportation task be attached to agents that plan for it and perform it. Therefore, our simulated transportation system is significantly different from DVMT.

The tileworld model [10] was used as a testbed for planning and task allocation and execution in multi-agent systems. The utilization of physics methods allows for a model that is significantly richer than the tileworld model. The tileworld model distinguishes (at least) two different procedures – deliberation and path planning – which are usually performed sequentially, whereas in the physics-based model an inherent property is interleaving planning and execution. And, while the tileworld proves to work successfully for systems of dozens of tasks and agents, (15 agents, 80 tasks in [6]), its computational complexity[6] will probably disable scaling up to thousands of tasks and agents. Such system size is allowed by the physics based model, as our simulations prove.

Ephrati, Pollack and Ur [6] suggest the multi-agent filtering strategy as a means for coordination among agents. They have conducted several experiments that show, that for the tile-world, this strategy improves the performance of the agents. This coordination is achieved without explicit negotiation. In our work we do not suggest a strategy, rather we suggest a method for modeling the goal-agent environment. Based upon this model we suggest a detailed algorithm for the single agent for acting efficiently in the environment.

Glance and Huberman [7] present a detailed physical formalism of the dynamics of the collective action of a system of individuals. In our work the main issue is the physical behavior of the single agent. Shoham and Tennenholtz [13] presented results of simulations that were performed in order to perceive the emergence of conventions in multi-agent systems. In our research, we discuss emergent cooperation and determine the social laws to be such – –physical laws– – that they will cause the emergent cooperation of the system when this cooperation is necessary. Mataric [9] proposes defining a set of basic interactions that will allow the simplification of group behavior analysis. In our work, we concentrate on the nature of the basic interactions and adopt the physical interactions among particles to model the interactions among agents and goals.

6 Conclusion

The problem of the behavior of agents in very large agent-societies imposes difficulties that are hard to solve even when the proposed solutions are of low-order polynomial complexity. The approach which we present suggests a solution to some aspects of this problem. We provide a method for task allocation which is applicable to several classes of large-scale cooperative MAS. The physics-based approach we present results in complexity which is, on the side of the single agent, very low and may even be $O(1)$. Such results are possible since we use a model whose behavior is already known.

[6] As Kinny and Georgeff [8] explicitly say: "to reduce the complexity...we employed a simplified Tileworld with no tiles."

Therefore, we are not required to perform the numerous explicit calculations that would have otherwise been necessary.

The model used and the algorithm that enables the single agent to act according to the model result in agents allocating themselves to goals in order for these to be satisfied. The agent-goal matching is an emergent result of the physics-oriented behavior of the agents. In cases where too many agents fit the requirements of the same goal, our model will disenable some of them from reaching the goal, via mutual rejection. As we have shown, our algorithm converges to a solution within reasonable time and leads to agent-goal allocation and execution. Our method does not lead to the optimal allocation, but reaching an optimal allocation requires complete on-line information about all of the agents and goals comprising the system and, for a large class of problems, an exponential computation-time.

Our model can rely on theoretical and experimental results that are already known from physics. Nevertheless we have performed simulations which support the theoretical observations. According to results from physics, we can predict the evolution of the modeled agent-system, since it should evolve in the same manner as a corresponding physical system. The local interactions, which enable one to derive the global behavior of the system, assure a low computational complexity of the model. In very large-scale agent-systems, this approach provides a model that allows for emergent cooperative goal-satisfaction activity, as shown in our experiments.

References

1. Keith Decker and Victor Lesser. Designing a family of coordination algorithms. In Victor Lesser, editor, *Proceedings of the First International Conference on Multi–Agent Systems*, pages 73–80, San Francisco, CA, 1995. MIT Press. Longer version available as UMass CS-TR 94–14.
2. E. H. Durfee. *Coordination of Distributed Problem Solvers*. Kluwer Academic Publishers, Boston, 1988.
3. E. H. Durfee, V.R. Lesser, and D. D. Korkill. Coherent cooperation among communicating problem solvers. *IEEE Transactions on Computers*, 36(C):1275–1291, 1987.
4. E. Durfee and V. Lesser. Predictability vs. responsiveness: Coordinating problem solvers in dynamic domains. In *Proceedings of the Seventh National Conference on Artificial Intelligence*, pages 66–71, St. Paul, MN, August 1988.
5. Edmund H. Durfee and Victor R. Lesser. Partial global planning: A coordination framework for distributed hypothesis formation. *IEEE Transactions on Systems, Man, and Cybernetics*, 21(5), September 1991. (Special Issue on Distributed Sensor Networks).
6. E. Ephrati, M. Pollack, and S. Ur. Deriving multi-agent coordination through filtering strategies. In *IJCAI95*, pages 679–685, 1995.
7. N. S. Glance and B. A. Huberman. The outbreak of cooperation. *Journal of Mathematical Sociology*, 17(4):281–302, 1993.
8. D. N. Kinny and M. P. Georgeff. Commitment and effectiveness of situated agents. In *Proceedings of the Twelfth International Joint Conference on Artificial Intelligence*, pages 82–88, Sydney, Australia, August 1991.
9. M. J. Mataric. Kin recognition, similarity, and group behavior. In *Proc. of the 15th Cognitive Science Society Conf.*, pages 705–710, 1993.

10. M. E. Pollack and M. Ringuette. Introducing the tileworld: Experimentally evaluating agent architectures. In *Proc. of AAAI90*, pages 183–189, Boston, MA, 1990.
11. D. C. Rapaport. Large-scale molecular dynamics simulation using vector and parallel computers. *Comp. phys. reports*, 9(1):1–53, 1988.
12. O. Shehory and S. Kraus. Cooperation and goal-satisfaction without communication in large-scale agent-systems. In *Proc. of ECAI-96*, pages 544–548, Budapest, Hungary, 1996.
13. Y. Shoham and M. Tennenholtz. Emergent conventions in multi-agent systems: initial experimental results and observations. In *Proc. of KR-92*, 1992.
14. R. G. Smith. The contract net protocol: high-level communication and control in a distributed problem solver. *IEEE Transaction on Computers*, 29(12):1104–1113, 1980.
15. C. Trozzi and G. Ciccotti. Stationary nonequilibrium states by molecular dynamics. II. Newton's law. *Phys. rev. A*, 29(2):916–925, 1984.
16. M. Wellman. Market-oriented programming: Some early lessons. In S. Clearwater, editor, *Market-Based Control: A Paradigm for Distributed Resource Allocation*. 1995.
17. M. P. Wellman. A market-oriented programming environment and its application to distributed multicommodity flow problems. *Journal of Artificial Intelligence Research*, 1:1–23, 1993.
18. M. Yokoo, E. Durfee, T. Ishida, and K. Kuwabara. Distributed constraint satisfaction for formalizing distributed problem solving. In *Proceedings of the Twelfth International Conference on Distributed Computing Systems*, pages 614–621, 1992.