Behaviosites: Manipulation of Multiagent System Behavior through Parasitic Infection

Amit Shabtay and **Zinovi Rabinovich** and **Jeffrey S. Rosenschein**
School of Engineering and Computer Science
The Hebrew University of Jerusalem
Jerusalem, Israel
ashabtay@gmail.com, {nomad, jeff}@cs.huji.ac.il

Abstract

In this paper we present the Behaviosite Paradigm, a new approach to coordination and control of distributed agents in a multiagent system, inspired by biological parasites with behavior manipulation properties. Behaviosites are code modules that "infect" a system, attaching themselves to agents and altering the sensory activity and actions of those agents. These behavioral changes can be used to achieve altered, potentially improved, performance of the overall system; thus, Behaviosites provide a mechanism for distributed control over a distributed system. Behaviosites need to be designed so that they are intimately familiar with the internal workings of the environment and of the agents operating within it.

To demonstrate our approach, we use behaviosites to control the behavior of a swarm of simple agents. With a relatively low infection rate, a few behaviosites can engender desired behavior over the swarm as a whole: keeping it in one place, leading it through checkpoints, or moving the swarm from one stable equilibrium to another. We contrast behaviosites as a distributed swarm control mechanism with alternatives, such as the use of group leaders, herders, or social norms.

Introduction

Nature presents us with a wealth of functioning problem solutions (Vincent & Mann 2002), available for inspiration and adaptation to technological contexts. The concept of "parasite", for example, taken from biological systems, has been adopted (in multiple ways) by computer science researchers. The notion of one agent exploiting other agents has been used in simulations of evolution (Ray 1990), and their damage to the host has been harnessed as a driving force of selection in genetic algorithms (Hillis 1992). Parasites have also provided the common name given to computer malware.

However, one of the most fascinating properties of parasites in nature has not received attention in the computer science community, namely, the ability of these (usually microscopic) entities to control behavioral features of their (usually macroscopic) hosts. Such is the case, for example, with rabies, in which parasites cause the infected animal to act aggressively, aiding their kin transmission to the next host. In this paper, we describe an approach to enabling one agent to modify the behavior of another agent in a multiagent system (MAS). We will define the **behaviosite concept** — parasitic agents that facilitate behavioral changes in their hosts, to achieve a global system effect.

Copyright © 2006, American Association for Artificial Intelligence (www.aaai.org). All rights reserved.

Distributed System Control and Coordination

The challenge of developing sophisticated control and coordination mechanisms for societies of agents, so as to achieve desired system-wide behavior, is a central theme of MAS. There are many proposed solutions, including social norms (Shoham & Tennenholtz 1992), Generalized Partial Global Planning (Decker & Lesser 1992), hierarchical organizations, and agent roles/functions.

The Behaviosite Paradigm that we present is another way of engendering certain kinds of behavior in a multiagent system. Each behaviosite manipulates the behavior of its current host so that the overall effect of all these behavioral changes, engendered by all behaviosites, results in certain system-wide behavior. Behaviosites in this model are not independent agents, but rather are dependent on their hosts. Moreover, behaviosites act within the functional limits of their host and environment, and do not add new sensing or action abilities to their hosts.

The control ability of behaviosites will be demonstrated on a swarm of *floys* (Dolan 2005) — agents, each of which is governed by rules ensuring that it is not too close to other agents, nor too far. These rules give rise to behavior similar to a swarm of flies or a flock of sheep. With simple yet effective infection and manipulation strategies, and only a small percentage of floys infected, we show that behaviosites can make the swarm stay in one place, move along a predefined path, or overcome attractors. Since behaviosites are endemic in the swarm, and robust to failures of some of the behaviosites, they have advantages over the use of leaders or a hierarchal structure. They are also well-suited to computer graphics simulations, since despite their control over the swarm, the behavior of the swarm remains life-like.

In the next section we present a description of the Behaviosite Paradigm. We then discuss some swarm control mechanisms, and show how behaviosites manipulate the swarm to achieve several tasks. In the section on related work, we briefly overview the concepts of motion planning, adjustable autonomy, and mechanism design. We conclude with a discussion of our approach and of future work.

The Behaviosite Paradigm

Behaviosites are a special kind of agent that achieves a global effect on the behavior of the system through local effects on the behavior of individual agents. A behaviosite uses *system specific* information in affecting the behavior of an agent, so that the overall system dynamics will change to some other stable system dynamics.

The basic system is composed of agents and an environment. Behaviosites, by definition, parasitize an entire functional system, and are a property of neither the environment nor of agents. Their individual performance is not what matters, but rather their effect on the system as a whole.

System Requirements

For the Behaviosite Paradigm to be viable and successful, some system requirements must be met.

Multiple agents: By definition, a behaviosite manipulates the behavior of one agent; if there is only one agent in the system, one could simply create a new behavior module for the desired effect. In a society of agents, behaviosites' mobility has advantages over the insertion of behavioral modules into all the agents. For example, a behaviosite's infection strategy can be designed to infect the most suitable agents in the society — perhaps the weakest, or the strongest, links. As agent roles may change over time, behaviosites can dynamically choose better agents to infect.

It should also be noted that a multiagent system usually has changeable system dynamics — an implicit requirement for this paradigm.

Susceptible agents: Behaviosites alter the parasitized host behavior, by taking away some of its autonomy; the behaviosite must have some method of affecting agent behavior. It may either be through the use of an existing "hook" in the agents, that allows access to the latter's internal mechanisms, or it may be via some external environmental effect. For example, a robot moving forward might deviate from its course either via an internal behaviosite (that changes its internal logic), or via an external behaviosite manipulating the robot sensors' input, so that the robot senses a false obstacle. We discuss this "internal/external" distinction in more detail below.

Transfer Medium: Behaviosites can infect different hosts over time, and must thus have some "transfer medium" between hosts. The medium can be built into agent-to-agent interactions, allowing (for example) behaviosites to travel via inter-agent communication. The medium can also be implicit, such as an environment that actively creates behaviosites ex-nihilo — the behaviosites are "spawned" by an environmental mechanism, and infect agents directly via the environment (different agents, over time).

Behaviosite Requirements

Benefiting the system: Behaviosites are not required for normal system performance, but should be designed to be beneficial to the system in some respect. Benefit to the system may be increased social utility, or altered performance (even causing the system to exhibit new capabilities, as shown in our parasitized floys example below). A behaviosite need not be beneficial to a host agent — it may even harm the host. Moreover, a behaviosite need not be beneficial to itself (which differs from parasites in nature).

Deep system knowledge: For the behaviosite to successfully affect the agent and, through that action, the entire system, the behaviosite designer must employ deep system knowledge. He must find ways to overcome possible resistance or over-sensitivity to system change, so that the effect on infected agents will spread throughout the system, and not be suppressed by, say, learning agents.

Use existing capabilities: Behaviosites only augment existing capabilities of the infected agents. They can add new system capabilities, by using existing agent ones. The behaviosite simply uses the agent's architecture to achieve global system change.

Small numbers: Since behaviosites come with costs (design time, run-time, building costs) we would like to have the minimal number of behaviosites that can achieve a desired goal. Deep system knowledge has a significant effect on the number of behaviosites needed.

Mobility between hosts: Moving from host to host enables the behaviosite to position itself in the most effective place at a given time. The infection strategy by which behaviosites mobilize themselves is of no less importance than the manipulation strategy, since not being at the right place at the right time can render behaviosites ineffective.

Behaviosite Location in the System Flow

The behavior of an agent can be manipulated using a variety of techniques, falling into two main categories — external and internal behaviosites. External behaviosites can affect the input or output of the parasitized agent (Figure 1a). Input can be manipulated by blocking some of the host's sensors, or providing false sensory input. They can affect an agent's output behavior by altering actions, e.g., by distorting communication between the agent and other agents.

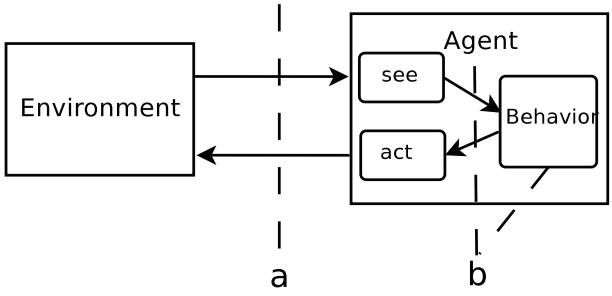

Figure 1: (a) marks where an external behaviosite can alter the host's behavior, by altering its input/output. (b) marks where an internal behaviosite can alter the host's behavior, by altering internal data or by partial or full replacement of behavior modules.

Internal behaviosites (Figure 1b) are more versatile in their methods of manipulation. They can manipulate internal data — changing the beliefs, desires, or intentions of the

agent, so the agent will come to decisions based on manipulated data. Another method of manipulation is distorting or changing the translation of the input from the sensors to beliefs, or altering the decision reached by the logic module, thus resulting in a different output action than the logic module intended. The most extreme manipulation of an internal behaviosite is modifying, or completely replacing, its host's logic module. However, since the behaviosite's influence is temporary, the agent will regain autonomy as soon as the behaviosite leaves.

External behaviosite manipulation does not require any access to the internal workings of the agent, and as such, can be employed in an already built and functioning system. On the other hand, internal behaviosites require that host agents have hooks that will allow for sensitive internal manipulation. The incentive for agent designers to prepare these hooks does exist, especially in a cooperative society where behaviosites may facilitate better cooperation. Such incentives can even exist in a society composed of self-interested agents, as shown in (Shabtay, Rabinovich, & Rosenschein 2006). If the system requires the existence of these hooks in participating agents, or if it can be guaranteed that every individual agent will benefit from the behaviosite, there is motivation to include these (sufficiently protected) hooks.

Optional Behaviosite Features

Apparent vs. hidden infection: Behaviosites may "tag" an agent by infecting it, and this tag will facilitate behavioral changes in the surroundings of the agent (such as making other agents avoid contact with this agent). However, hidden infection has its own merits, e.g., preventing other agents from exploiting the fact that a fellow agent is infected.

Inter-behaviosite communication: Behaviosites can form a network within a single agent and achieve better results by augmenting one another, or by having several complementary effects. Another use of communication is the creation of a society of behaviosites within a society of agents, for better coordination within the agent society.

Controlling Swarm Motion with Behaviosites

To demonstrate the concept of behaviosites, we applied them to the domain of swarm motion. Controlling a swarm of robots (known as large group motion planning) has received a great deal of attention, since there are many applications for these techniques (including, but not limited to, military settings). Imagine that we want to move a swarm from one coordinate to another, possibly going through several checkpoints along the way. Controlling such a mass of robots requires sophisticated solutions, since it is not feasible for a single agent to guide each robot separately towards the goal coordinate, without losing robots along the way, or losing the compactness of the group.

Reynolds (Reynolds 1987) showed that it is possible to generate realistic bird flocking motion as an emergent behavior when each agent, called a boid, obeys three very simple rules: 1. *Separation*: steer to avoid crowding local flock mates; 2. *Cohesion*: steer to move toward the average position of local flock mates; 3. *Alignment*: steer toward the average heading of local flock mates. Each boid has a limited field of view, so it cannot know the position or alignment of all group members.

The first two rules are essential for flocking behavior. However, when the third rule is removed, the emergent mass behavior is like a swarm of flies (or sheep, depending on agent speed); thus, an agent with only the first two rules is termed a floy. An individual floy's motion appears to be "independent" (looping, zig-zagging like a fly), but the swarm's cohesion is maintained.

General Swarm-Behaviosite Description

Without any rule specifying direction, floy swarms tend to move in a moderately coherent form erratically in space, without any directionality. This serves as a convenient substrate for controlling the motion of the swarm towards desired routes, by exploiting the second rule of floys.

Behaviosites serve as a distributed control mechanism to move the swarm from one checkpoint to another, using several non-communicating behaviosites and only sensing the local environment (for simplicity, behaviosites have the same sphere of sensing as the floy). Behaviosites are not a special type of floy, and floys do not sense them. They act as a "swarm within the swarm" — infecting swarm members by jumping from one floy to another and making them steer toward the current coordinate that the behaviosites desire.

These behaviosites embody our general requirements. Behaviosites facilitate the creation of new capabilities of the swarm (the swarm's ability to move in a certain path), and thus benefit the system. To do so, they use deep system knowledge: exploiting the floy design rules and sensors, and using existing floy capabilities to achieve the desired motion. As we shall see, we will require only a relatively small number of behaviosites to achieve the effect, by having behaviosites move themselves to the floy that currently has the strongest effect over the swarm.

Creating a movement of the swarm in a desired path might seem easy at first — just steer some of the floys in the desired direction, and the whole swarm will follow. However, there are several issues that require attention. We want to: 1) move the swarm without breaking it into subgroups, having no knowledge of the location of all members; 2) use as few behaviosites as possible; and 3) know when the swarm has reached a checkpoint so that the behaviosites can guide it toward the next checkpoint, without any communication among them.

Three main high-level design questions must be answered: 1) external vs. internal behaviosites (and hence do they affect input, output, logic, perception of environment, decided action); 2) possible modes of transmission; 3) possible effects on infected floys (manipulation). In our swarm model there were several possibilities for infection and manipulation. We used external behaviosites that traveled by jumping from floy to floy. Movements of floys are done in "turn units"; each time step, a floy can turn one turn unit. In our simulation, the behaviosites' effect is only in manipulating the turns of the floys, by applying force that causes (two, if not specified otherwise) turn units toward the desired checkpoint, in addition to the one decided by the floy.

Behaviosites' Internal Logic

After deciding on the high level design questions, and having only limited information regarding the environment, one must decide on the internal logic of the behaviosite (its specific infection and manipulation strategies), as well as problem-specific questions (such as how to decide that the swarm has reached a certain checkpoint, and steer it to the next one).

Infection Strategy As mentioned above, in order to use as few behaviosites as possible we would like behaviosites to infect the most influential floy at a given time step. Since floys move toward the sensed center of mass, the floy with the most neighbors is the most effective. However, behaviosites do not have that kind of information. They only know what their host floy may know — the position of close floys and their orientation. One of the infection strategies that proved to be a failure in our experiments was the strategy of *infect forward* — infecting the floys closest to the next checkpoint. It caused the separation of the swarm into subgroups, and the separation of the behaviosites from the center of mass of the swarm.

A strategy of *infect backward* (infecting the floys furthest from the checkpoint) proved to be better. *Infect backward* had both the property of pushing the entire flock towards the checkpoint, and also picking up stray floys that separated from the main swarm mass, since they were usually the furthest from the checkpoint. However, the *infect backward* strategy did not have the desired property of infecting the floy with the most neighbors (the most influential floy). A combination of *infect backward* 80% of the time, and *infect forward* 20% of the time proved to be the most effective (using randomized "coin flips"). This strategy helped the behaviosite spend some of the time in the center of mass of the entire swarm without real knowledge of its whereabouts. They also stayed long enough on the outskirts of the swarm, preventing floys from separating from the swarm, and not too long in the front, which might have resulted in possible separation from the swarm. For simplicity, a floy could only be infected by one behaviosite at a time.

Moving between checkpoints After discovering (through analysis, and empirically) a good infection strategy, and fixing the manipulation strategy (two turn units toward the checkpoint), we wanted behaviosites to move the swarm from one checkpoint to another. For that, all behaviosites need to decide as simultaneously as possible if the whole swarm has reached a checkpoint, and that it is time to steer it to the next checkpoint. As behaviosites travel "on board" floys, and not by themselves, they were not required to pinpoint a coordinate, but rather a small circular area around the checkpoint coordinate. Because they travel back and forth in the swarm, when the swarm is approximately in the right position, it takes a relatively short time for all behaviosites to discover that they are at the checkpoint.

However, if the swarm travels fast (depending on the distance between checkpoints, and the percentage of behaviosites), there are times when not all behaviosites update themselves; they do not realize that the swarm has reached the checkpoint and that they need to steer it to the next one. This may cause the swarm to split or become indecisive, since behaviosites pull toward different directions. Because of that, another heuristic was introduced. When a behaviosite discovered it was at a checkpoint, regardless of the checkpoint it was aiming at, it updated its beliefs accordingly and started steering the swarm toward the checkpoint following the one it was at (checkpoints had an order relation). So, without communication, and relying on the natural traits of the floy to stick with the swarm, behaviosites can correct their errors by themselves, leaving the system relatively robust to such problems.

Parasitized Swarm Simulation

To test the effect of behaviosites over the swarm, we tested several tasks using varying sizes of swarms and percentage of behaviosites. Uninfected floys were colored black, while infected ones were colored pink. The world was two-dimensional (700x600 pixels, width and height respectively), and behaviosites consider themselves at a checkpoint if they are at a distance of 20 pixels from it.

Three main tasks were explored — staying in one place, moving between checkpoints with varying curvature, and moving from one stable state of the swarm to another. Sizes of swarms were 5, 50, and 100, and the numbers of behaviosites were 1–10, and then in jumps of 10 — stopping at the maximal number of behaviosites, which is the number of floys for each swarm size. All tasks had a maximal completion time of 10,000 time steps. Each trial (n floys and m behaviosites) was repeated 20 times.

Staying in one place This task was meant to test the simplest behavior that behaviosites can add to the swarm, since no special coordination among behaviosites is needed. Since non-parasitized swarms move erratically in space, behaviosites must actively employ regulatory force to fix the swarm to one location.

From Figure 2a we see that the distance between the swarm center of mass and the single checkpoint drops exponentially as the percentage of behaviosites increases. We can conclude that for an infection rate (percentage of infected floys) of about 5%, the distance from the checkpoint can be considered sufficiently low. A swarm of 5 floys is not sensitive to such a percentage, but swarms of 50 and 100 floys exhibit almost identical behavior. In Figure 2b we see that the cohesion of the swarm also increases (the standard deviation (SD) of the swarm drops), but in a moderate manner. Pink floys are less affected by the first rule of being repelled by other floys, increasing cohesion. However, as long as not all are infected, there are some floys obeying this rule and being repelled from the swarm mass. This explains the moderate increase in cohesion.

Coping with Several Checkpoints After establishing the fact that a handful of pink floys can indeed affect the global behavior of the swarm, we tested if they could also make it follow, as fast as possible, a desired path marked by checkpoints, with as few behaviosites as possible and without breaking the swarm.

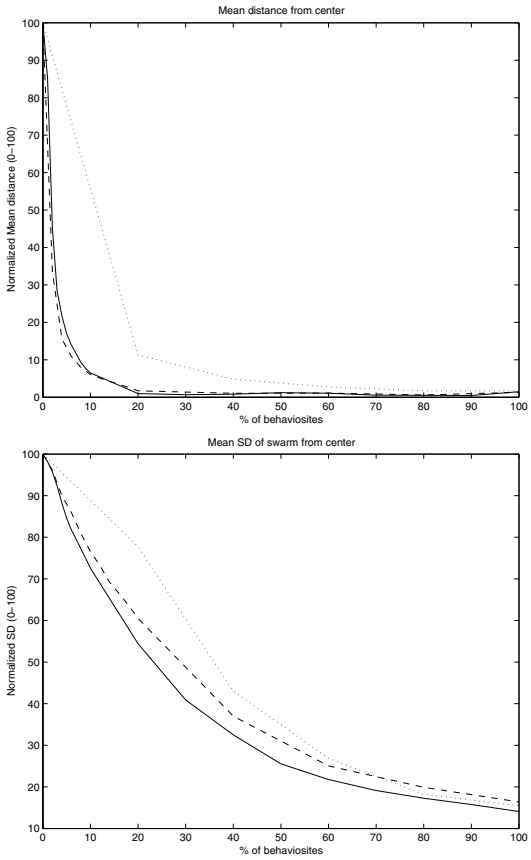

Figure 2: Behavior of the swarm in a mission to stay at one checkpoint, as a function of percentage of infected floys. (a) Normalized mean distance of swarm center of mass from single checkpoint. (b) Mean SD of the swarm mass — cohesion. Legend: ...: 5 floys, - -:50 floys, —:100 floys

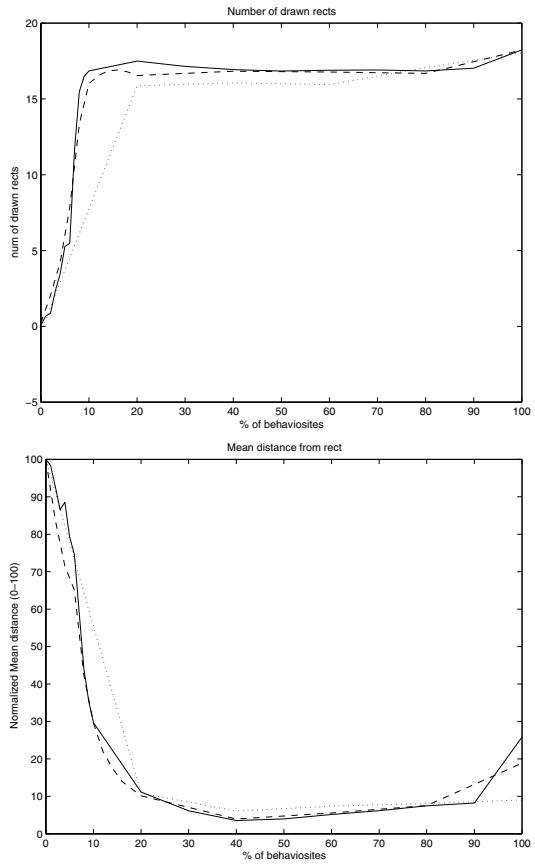

Figure 3: Behavior of the swarm in a mission to follow four consecutive checkpoints forming a rectangle in a circular manner, as a function of percentage of infected floys. (a) Mean number of drawn rectangles during 10000 time steps. (b) Normalized mean distance of swarm mass center from the path of the rectangle. Legend: ...: 5 floys, - -:50 floys, —:100 floys

Behaviosites were given several consecutive checkpoints to follow, pulling the swarm along, without communication among themselves. They were to follow, repetitively, the path marked by checkpoints, and after 10000 time steps, the number of repetitions and distance from the desired path were checked. Figure 3 describes a mission of following four checkpoints forming a rectangle. The performance increase is also exponential in the percentage of behaviosites and we reach maximum performance for an infection rate of 10%. Again, an infection rate of about 5% is sufficient to create a reasonable movement along the desired path.

The normalized mean distance from the rectangular path (Figure 3b) behaves similarly to that of one checkpoint, with one major exception. For a very high infection rate (about 90%), the pink floy swarm members not only do not properly obey the first rule of repulsion from one another, but also do not obey the second rule of cohesion. Every pink floy to himself — they break apart from the swarm.

Testing the same experiment for a circular shape (8 vertices along a circle) gave similar results.

Moving from One Equilibrium Point to Another Sometimes the task of the behaviosites is to shift the society's equilibrium from one point to another. To test that with the swarm model, we considered the following problem. On each side of the world there is an attractor for the floys — one at the middle top and one at the middle bottom. In addition to the turn unit that a floy decides upon, the attractor causes the floy to exert one more turn unit in the direction of the relevant attractor. If positioned in the upper (lower) half of the world, the floy is attracted to the upper (lower) attractor. The two attractors present two equilibrium points in the world — each in itself is stable, but moving from one to another is a hard task. The task of the behaviosites is to pull the entire swarm from the upper attractor to the lower one. The main problem for the behaviosites is to overcome the enormous force that pulls the floys to the attractor — both the attractor and the cohesion force. For that, each behaviosite uses five turn units toward the lower attractor.

In Figure 4a we can see that moving from one attractor

to another is almost a step function. Behaviosites either succeeded or failed, and if successful, it did not take long to bring the swarm to the second attractor. We also note that it took a relatively high infection rate of about 20% to achieve the goal of moving the swarm. However, this infection rate almost matches the forces applied on floys. Since 20% of pink floys pulling toward the second attractor use the same force as 100% of floys pulling toward the first attractor, the swarm was supposed to stay at the first attractor, but the movement dynamics created by the behaviosites were probably sufficient to succeed in the mission of pulling the swarm away.

Now, even though behaviosites can apply enough force to move the swarm, we faced another problem — not leaving floys behind in the process, and not splitting the swarm in the middle point between the two attractors. For that, the infection strategy of *infect backward* combined with *infect forward* proved to be most successful. Floys left behind were usually immediately infected and pulled back toward the center of the swarm, since they were the furthest from the second attractor. This is especially true when crossing the middle point between the attractors. However, there were cases in which some floys were left behind, without any behaviosite to pull them, as can be seen in Figure 4b (the completeness problem) — but usually behaviosites succeeded in bringing all swarm members to the second attractor.

Advantages of Using Swarm Behaviosites

We briefly list some of the advantages of using behaviosites in a swarm of floys without any communication or explicit coordination: 1. Can create a movement of the swarm along a path; 2. Is robust to malfunctioning, ill-functioning, or destroyed behaviosites, since they are completely distributed, and the effect of such loss of behaviosites can be known in advance (and compensated for); 3. Behaviosites are endemic, thus protected by the swarm from external harm; 4. Few can control many; as the most effective "leader" keeps changing, so does the position of the behaviosites in the swarm; 5. They can move to the most effective position at a given time without disturbing the swarm.

Related Work

Multiagent Motion Planning

The classical approaches for multiagent motion planning are centralized (dimensionality problem for large groups) vs. decentralized planning (completeness problem) (Latombe 1991). There is also research that uses external agents to control the steering of the flock. One is multi-shepherding (Lien *et al.* 2005) — a distributed control mechanism to move around a flock of sheep, using several non-communicating shepherds that use a repelling field. Another (Chaimowicz & Kumar 2004) tackles the problem of controlling a swarm of unmanned ground vehicles (UGVs) using unmanned aerial vehicles (UAVs). They divide agents (UGVs) into smaller groups, to reduce the dimensionality problem by semi-centralized motion planning (using UAVs). This hierarchical structure demands handling robustness issues, in the case of local UAV failure.

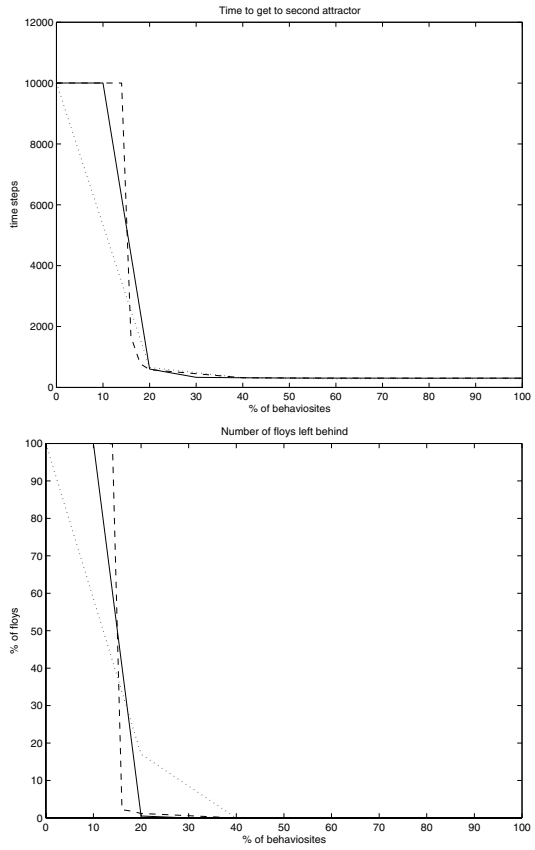

Figure 4: Behavior of the swarm in a mission to move from one attractor to another, as a function of percentage of infected floys. (a) Time steps to reach second attractor, with 10000 as a limit for failure (started pulling only after 100 time steps). (b) Number of floys left behind, in the first attractor. Legend: ...: 5 floys, - -:50 floys, —:100 floys

Mechanism Design and Adjustable Autonomy

Mechanism design is another alternative when we wish to bring about certain behavior in a society of agents. It addresses the problem of designing a distributed protocol that will implement a particular objective despite the self-interest of individual agents (Dash, Jennings, & Parkes 2003). In contrast, behaviosites are not a protocol that agents might rationally choose to obey, and do not provide an "incentive" for agents, so that a certain behavior pattern will be preferred. Instead, behaviosites, like parasites in nature, utilize agents' existing tendencies to gain an overall system effect.

There are similarities between the effect of behaviosites on a host, and the concept of adjustable autonomy (AA) (Scerri, Pynadath, & Tambe 2002). AA is concerned with changing the level of autonomy of an agent to improve system performance when such adjustment is warranted (control is ceded to another agent or to a human). A similar rationale (and analogous mechanism) exists with Behaviosites. An agent might (unwittingly) cede some of its autonomy to behaviosites that make local behavioral changes, usually with only partial information, so that the

entire system's behavior is changed. The practical manifestation of AA is usually in agent-human synergy, while behaviosites live in an a more agent-centric world.

Control of Macro-Spatial Structures

In their work, Mamei, Roli and Zambonelli (Mamei, Roli, & Zambonelli 2005) presented an idea closely related to that of the Behaviosite Paradigm, within the context of cellular automata. They presented a method for controlling the global behavior of cellular automata using local influences. In their approach, a subset of the cellular array was selected on which to introduce a beneficial signal, either compatible or destructive of some global pattern. This signal was then diffused within the cellular system due to asynchronous time lines and globally effective random perturbation. The Behaviosite Paradigm can be viewed as having a similar structure — behaviosites infect an agent (with an infection strategy more sophisticated than just perturbation), and change its output behavior (the "special" cellular automata). However, Behaviosites' effect is local in time and space, in contrast to what is discussed in (Mamei, Roli, & Zambonelli 2005). The former also do not rely on an existing diffusion effect, but rather rely on their mobility and infection strategies to seek key control points.

Mamei et al. maintain, as do we, that in the near future there will exist more systems characterized by distribution and autonomy, operating within highly dynamic environments. These systems will by their nature require decentralized control mechanisms, such as those presented in this paper and in theirs.

Discussion and Future Work

The core of the Behaviosite Paradigm is creating distributed behavioral changes in a small number of agents (by taking advantage of existing capabilities) using parasites, so that the sum of the distributed actions of all agents will lead to different system behavior. These parasites are not part of the society, and they act as a society within the society.

In this paper, we presented the Behaviosites Paradigm and a specific application for it — a control mechanism over a swarm. We specified the characteristics of systems in which this paradigm can be employed and behaviosites' required traits, which are necessary for their system integration and effectiveness. We also described two types of behaviosites — internal and external, depending on whether agents were originally designed to accommodate them or not.

We then described a specific system in which behaviosites are effective. Controlling a swarm of robots is now of practical concern (Chaimowicz & Kumar 2004). However, even in everyday systems (such as massive role playing games on the web, or simulated creatures in computer graphics), the behaviosite solution is highly effective. Controlling the many by the few in a robust manner without losing the basic characteristics of the swarm is desirable in all of the above systems.

The potential strength of the Behaviosite Paradigm remains to be explored. In the future we would like to examine more systems in which the Behaviosite Paradigm is effective. One example is the automatic creation of stories, an evolving field, which has gained much attention due to the high potential of its usage in computer games.

Acknowledgment

This work was partially supported by grant #039-7582 from the Israel Science Foundation.

References

Chaimowicz, L., and Kumar, V. 2004. A framework for the scalable control of swarms of unmanned ground vehicles with unmanned aerial vehicles. In *Proceedings of the 10th International Conference on Robotics and Remote Systems for Hazardous Environments*.

Dash, R. K.; Jennings, N. R.; and Parkes, D. C. 2003. Computational mechanism design: A call to arms. *IEEE Intelligent Systems* 18(6):40–47. (Special Issue on Agents and Markets).

Decker, K. S., and Lesser, V. R. 1992. Generalized partial global planning. *International Journal of Intelligent Cooperative Information Systems* 1(2):319–346.

Dolan, A. 2005. Floys: Overview. http://www.aridolan.com/ofiles/Floys2.html.

Hillis, W. D. 1992. Co-evolving parasites improve simulated evolution as an optimization procedure. *Artificial Life II*.

Latombe, J. C. 1991. *Robot Motion Planning*. Kluwer Academic Publishers.

Lien, J.-M.; Rodriguez, S.; Malric, J.-P.; and Amato, N. M. 2005. Shepherding behaviors with multiple shepherds. In *Proceedings of the 2005 IEEE International Conference on Robotics and Automation (ICRA)*.

Mamei, M.; Roli, A.; and Zambonelli, F. 2005. Emergence and control of macro-spatial structures in perturbed cellular automata, and implications for pervasive computing systems. *IEEE Transactions on Systems, Man, and Cybernetics, Part A: Systems and Humans* 35(3):337–348.

Ray, T. 1990. Tierra. http://www.his.atr.jp/∼ray/tierra.

Reynolds, C. W. 1987. Flocks, herds, and schools: A distributed behavioral model. *Computer Graphics* 21(4):25–34. (SIGGRAPH '87 Conference Proceedings).

Scerri, P.; Pynadath, D. V.; and Tambe, M. 2002. Towards adjustable autonomy for the real world. *Journal of Artificial Intelligence Research* 17:171–228.

Shabtay, A.; Rabinovich, Z.; and Rosenschein, J. S. 2006. Behaviosites: A novel paradigm for affecting distributed behavior. In *The Fifth International Joint Conference on Autonomous Agents and Multiagent Systems*. To appear (short paper).

Shoham, Y., and Tennenholtz, M. 1992. On the synthesis of useful social laws for artificial agent societies. In *Proceedings of the Tenth National Conference on Artificial Intelligence (AAAI-92)*.

Vincent, J. F. V., and Mann, D. L. 2002. Systematic technology transfer from biology to engineering. *Phil. Trans. R. Soc. Lond. A* 360(1791):159–174.