

Towards Intelligent Data-Driven Simulation for Policy Decision Support in the Social Sciences

Catriona Kennedy and Georgios Theodoropoulos
School of Computer Science, University of Birmingham, UK
{cmk,gkt}@cs.bham.ac.uk

Abstract

In the physical sciences, dynamic data driven simulation (DDDAS) is used to absorb new data into a simulation of the observed system (e.g. the atmosphere) causing the simulation states to be continually adjusted. The underlying model on which the simulation is based may be revised as a result of data assimilation. Conversely, the simulation predictions may play a role in selecting the data to be absorbed.

In this report we discuss the challenges of applying DDDAS to agent-based simulations in the social sciences, with the aim of providing reliable “what-if” scenarios for policy decision support (e.g. prediction of the impact of a particular policy). Models of social systems are often very simplified, meaning that data-driven adaptation is particularly essential to avoid making decisions based on flawed theories. We discuss the contribution that artificial intelligence can make to the design of a software agent to manage the DDDAS process. In particular, recent developments in cognitive agent architectures are already addressing many of the data-driven adaptation issues.

Key words: agent, cognition, decision support, fault-tolerance, simulation, social sciences.

1 Introduction - Dynamic Data Driven Simulation

In the physical sciences, “dynamic data driven simulation” (DDDAS) is a method where data from a physical system is absorbed into a simulation of the system[7]. The simulation typically represents a chain of events that is concurrently happening in reality (e.g. the atmosphere, a forest fire) and is used to make predictions about its future states. It can also be used for “what if” scenarios.

The predictions of the simulation are continually adjusted by absorption of new data. The underlying model on which the simulation is based may be revised as a result of data assimilation. It is also possible that the states predicted by the simulation should play a role in selecting the data to be absorbed. If such interactions between simulation and reality are to be automated, they will require intelligent software to manage them.

If DDDAS is used for an artificial system, its predictions may be used to influence the real physical system (for example, to optimise or adapt it). Data driven simulation that affects the physical system providing the data is called “symbiotic simulation” because of the mutual benefits of the simulation and the physical system on each other. Examples include semiconductor component testing [15].

In this paper we will discuss architectures for intelligent management of DDDAS with the aim of applying it to decision-support in social scenarios. Intelligent management involves making the following decisions (among others):

1. What new data should be absorbed, when and how? When should attention be focused on a particular data source for absorption into the simulation?
2. When should newly observed values that are outside of the normal range replace predicted ones in the same context and when should there be some revision of the model on which the simulation is based?

These two decisions are the basic requirements for a DDDAS system. We will consider how they can be automated (partially or fully) with artificial intelligence (AI) techniques.

1.1 DDDAS and Cognitive Agents

The process of predicting future states of the environment and continually adjusting them to new events is an important requirement of cognition. Simulation can be used as a form of “what if” reasoning for an intelligent agent (e.g. [24]). Such an agent could also control a data-driven simulation so that it assimilates aspects of the reality and revises existing models as necessary.

The degree and type of learning and revision may vary. There is a spectrum of possibilities ranging from purely “hardwired” (little or no revision of initial model) to fully “discovered” (all concepts are created by interaction with the environment.) [28]. The reality will normally be somewhere in between, but the ability to dynamically vary the plasticity and focus of the learning processes is important.

We will use the term “DDDAS agent” is an abbreviation for what in reality will be a complex management system involving multiple agents (software and perhaps robotic) as well as Grid services.

1.1.1 Symbiotic simulation

If the DDDAS architecture is “symbiotic” in the sense defined in [15], this can also be modelled as a cognitive process. In natural cognitive systems, anticipation is used to direct perception and focus attention on a particular object (e.g. a cup on the table). The reality of the object can modify the further expectancy which may in turn direct attention to further objects or data sources that would not have been anticipated initially (e.g. if the cup is cracked or stuck to the table).

In a symbiotic simulation, predictions can be used as a basis for action on the observed system, just as anticipation does in a cognitive system. If direct action is not appropriate (e.g. because the observed system is not artificial) the predictions can be used to focus on relevant data sources to be assimilated. Such focusing is important, since otherwise the masses of potentially relevant data would become unmanageable.

Therefore, it is possible that research in cognitive agent architectures can make a valuable contribution to symbiotic simulation and vice versa.

1.1.2 Agent-based simulation

The world that the DDDAS agent is building a model of may include other “agents”. They can be natural or artificial systems (e.g. humans, or software). The agents may be modelled as components with objectively defined states only. In other words, they have no subjective beliefs or intentions, but just react to their environment. More complex models of agents have beliefs and affective states (such as anger) and may called “cognitively rich” [30, 25]. If social groups are being modelled, there may be a “prevailing” belief, set of values, goals or “typical” mood (e.g. the typical commuter travelling from work through the London tube). Organisations can be modelled as having goals, capabilities, authority etc.

[10] argues that a descriptive, symbolic approach to agent simulation can be more explanatory than a numerical one based on physics models. We will assume that both of these approaches can be combined. For example, the numeric approach can be used to model the physical environment of agents but a descriptive approach can be used for the agents’ decision-making processes and interactions among themselves.

Note that there is some overlap between the architectural requirements for a simulated agent and a software agent controlling the simulation. Cognition inspired architectures have some relevance to both on an abstract level. However, the role of emotion and “irrational” behaviour is only applicable to simulated agents. On the other hand a software agent would normally require a more detailed specification than a simulated agent.

1.2 Grid-based DDDAS and Autonomous Agents

It is important to compare the sort of Grid-based DDDAS agent that we are aiming for with a purely autonomous agent that uses internal simulation to predict states of its environment or its own components. An example is Nasa's Remote Agent [19, 21]. The agent monitors the spacecraft hardware, diagnoses faults and reconfigures the system as necessary. Model-based reasoning [8] is used to diagnose faults using an explicit model of the hardware based on state transitions. This model is the basis for internal simulations to predict the next state of components, given that they are working correctly. If the agent senses that the actual behaviour is different, it uses the dependencies in the model to identify the likely cause of the problem and plans a reconfiguration (see also general summary in [16], pages 298–305).

In theory, the Remote Agent could be extended to include adaptation and model revision, making it very similar to a DDDAS agent. The purpose of the DDDAS would be to serve the autonomy and robustness of the agent; the details of its operation (or of any model revision) could be hidden from any scientists who are end-users of the spacecraft (unless they want to know how the system is performing).

In contrast, most DDDAS systems are used to help humans understand a complex system and make appropriate decisions (not to help an artificial agent keep itself operational). The simulations are also much more complex than the those in a Remote Agent type of scenario. Figures 1 and 2 show this contrast.

1.2.1 Use of internal simulation by an autonomous agent

In Figure 1, note that the agent is “situated” in the world it is modelling: the world contains the software agent and the hardware it controls (e.g. it could be a robotic vehicle).

The simulation can be just a direct application of the model rules to the current state of the world to predict the next state (or states, depending on how far into the future the simulation runs). The simplest form of data assimilation is to take sensor readings after a simulation has run for a given number of time steps, re-initialise the current state with the new data and then resume the simulation. Clearly the new data may change subsequent predictions. In the Remote Agent example, the new sensor readings could be consistent with one of the failure modes in the model, which will lead to the prediction of subsequent failure states, thus requiring corrective action. More complex forms of assimilation can involve the continual absorption of data into a concurrently running simulation.

Sensors and effectors in the diagram are schematic and may involve software as well as hardware. The predicted state can determine what kind of data is important for subsequent assimilation and this requires direction of sensors. Sensor and effector activation can involve a complex translation between high level directions (thick arrows) and the actual low-level measuring or making adjustments to the hardware or environment (multiple thin arrows). Similarly the absorption of data (agent input arrows from sensors) can involve a non-trivial fusion and summarisation process in order to recognise what the current state is, or what known state is it closest to in the model. This also applies to a distributed Grid-based DDDAS system.

Model revision in a fully autonomous agent is ambitious (because of no human intervention). Therefore this action is labelled hypothetical in the diagram. Pure model-based reasoning does not involve the questioning of the model itself (e.g. if the data does not fit easily into any known failure state). We will discuss this limitation later. On the extreme end of the learning spectrum, the model would be developed entirely as a result of exploration and adaptation (but may not be feasible for realistic systems).

1.2.2 Assisted use of simulation in a scientific process

Figure 2 shows a scenario in which the DDDAS agent assists with a scientific process. Models and simulations have been developed separately for human understanding (labelled U in the diagram). In such an assistance scenario, the purpose of the simulation is primarily to help the scientist or other end

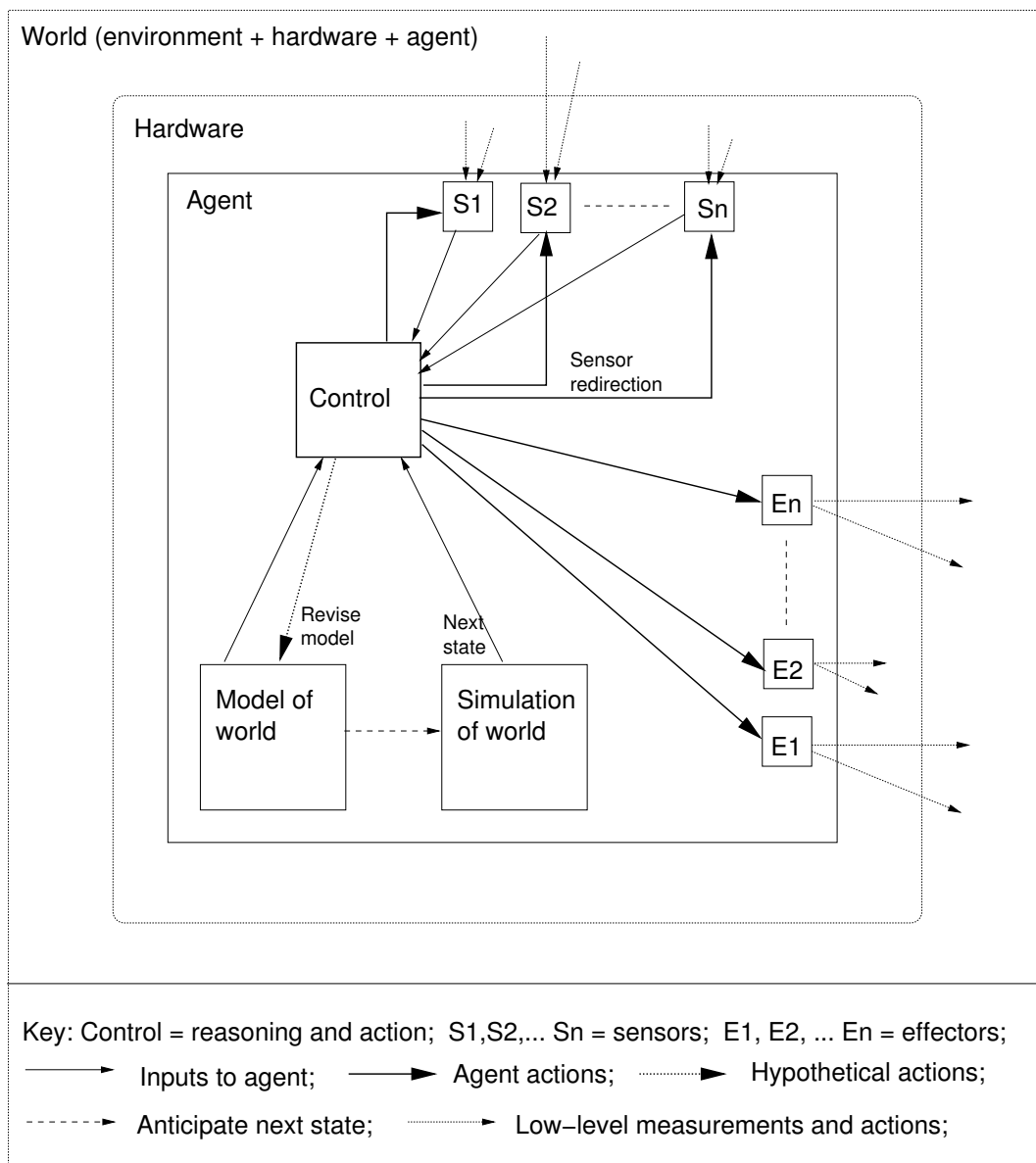


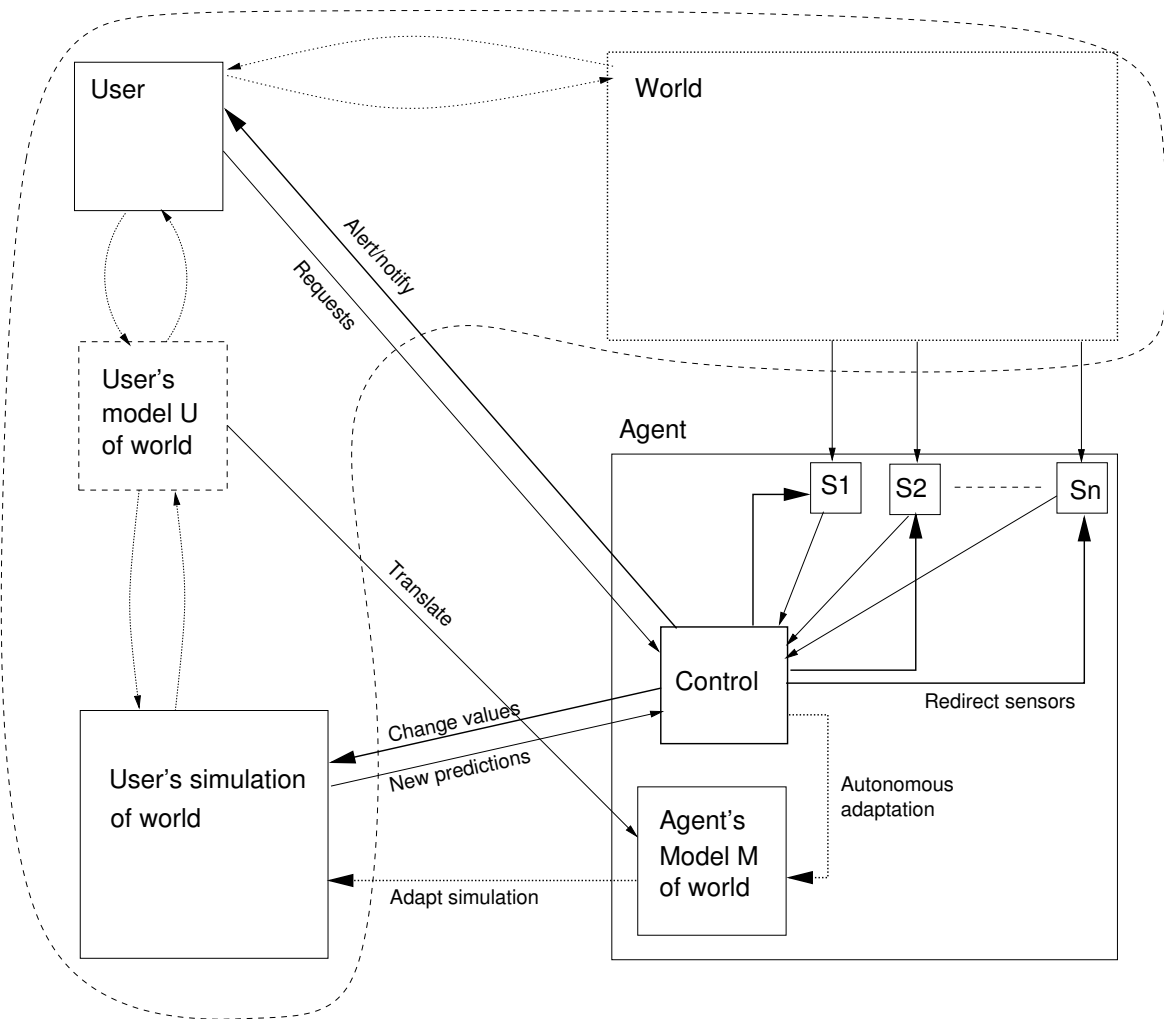
Figure 1: An autonomous agent using internal simulation and some DDDAS capability

user. It *may* also help the agent to adapt to its environment (and we will discuss later the advantages of this).

If the agent is to select the relevant data for absorption into the simulation, it must have some description of what is in the simulation and a representation of what the goals and priorities are. This description, which we can call D , can be a representation of the main entities and relations in the original model (U) in a form suitable for agent reasoning (e.g. it could be a set of rules or causal links). It can also act as the agent's internal model of the real world (labelled M in the diagram) which can evolve. For the moment we will assume that M is just D .

The nature of the relationship between the agent's internal model and the user's model depends on the application domain. For example, the user's model U could be a physics-type numerical model, while M contains semantic information about the entities that change state in the world. M may represent all or part of U . We can say that M contains "meta-data" about the user simulation, but also about the world itself.

The agent's model must distinguish between the user's simulation and reality. Effectively, the simulation is what the user model "says" about the world, while the "reality" is what the agent can



Key: Control = reasoning and action of agent; S1,S2,... Sn = sensors controlled by agent;

Inputs to agent;
 Agent actions;
 Hypothetical agent actions;

and Manual scientific process (experiment, theory formation, simulation).

Figure 2: A DDDAS agent assisting with modelling and simulation for a human user

find out independently by use of sensors and possibly effectors (sensors only are shown in the diagram). For example, each entity in the agent’s model could have a “simulated” version indicating its expected state in the future (according to the user’s model) and a “real” version indicating its observed states from sensor information (e.g. these could be numerical values of variables, actions of agents etc). Note that the user model’s predictions are not necessarily the same as what the agent might “anticipate” about the world based on its own model M , since this may evolve (see later).

For DDDAS, the agent needs to change the state of simulated entities to that of the real versions. One possibility is that the agent interacts with the simulation in the same way that a debugging system would interact with an application to inject values and “what-if” states into variables. Since the user simulation may include many details that are not represented in the agent’s model, the agent may only have partial control of it (i.e. it is a direct instantiation of U not M).

Model revision is more complex in the assistance scenario than in an autonomous agent and has to be done by interacting with the human user’s understanding of the simulation. One possibility is to include *provisional* model revision as a result of the agent’s independent interaction with the envi-

ronment. Two versions of the agent’s model might be used, one that remains static (D) and another version (M) that can undergo independent changes. (There may be several versions on different levels of abstraction as for the autonomous agent scenario). If there are substantial inconsistencies between the updated model and the original, the new model could be presented to the user as a suggestion for revision of the original.

The agent could also use M as a basis for internal data-driven simulation, just as in a Remote Agent type of scenario. This would be independent of the user simulation and could serve the purpose of the agent’s own adaptation and preliminary model revisions. In a more complex scenario, such internal simulations could occur on multiple levels of abstraction and include “meta-simulations” of the user simulation as well as simulations of the world. Such multi-level simulations are currently being developed in cognitive science [27, 12].

When considering AI contributions to DDDAS, we will be using the autonomous agent scenario as a starting point and then gradually adding the other components (such as humans in the loop etc.) and making modifications as necessary. Some application domains may not require significant modification.

2 Application Domains

In this section we look at a class of application domains in which a DDDAS agent acts as an “assistant” for a human decision-maker. In some cases, the agent may be called a “cognitive” assistant because it should remove some of the cognitive load from the decision-maker in situations where there is information overload and time sensitivity. We assume that such scenarios have the basic form of the science assistance scenario in Figure 2. However, scenarios that require less human intervention will have more of the properties of an autonomous agent.

Roughly, application domains can be divided into the following categories: natural systems, artificial systems, social systems. Although we are focusing on social systems it is important to look briefly at the other domains, so that they can be compared with the social domain.

2.1 Natural Systems

Many currently active DDDAS projects aim to help human experts understand and predict events in the natural environment. Although they involve purely numerical modelling, some projects are developing a Grid-based architecture that has much in common with the kind of intelligent management system that we are aiming for. The following are two examples.

[20] presents a DDDAS framework for the LEAD project (Linked Environments for Atmospheric Discovery) which is a developing infrastructure for real-time monitoring of weather events using distributed sensors. The data to be absorbed into a simulation is selected dynamically according to continually changing requirements. For example, there may be uncertainty about a weather event due to large disagreements between concurrent versions of a simulation (each with slightly differing physical parameters). In response, the DDDAS system should focus attention on the geographical region in which there is most uncertainty by re-directing the sensors so that they collect more data from this region, thus resolving the ambiguity.

Such dynamic focusing of attention is applicable to many other application domains. Another “universal” feature of the LEAD architecture is that the middleware providing the Grid computing resources is itself monitored and its resource allocations adjusted dynamically according to changing priorities in the application domain. This requires coordination and sharing of information across many boundaries which we will return to later.

The Poseidon project [18] aims to assist marine scientists to find relationships between physical/biological oceanography and ocean acoustics. Its DDDAS element includes two important features: *adaptive sampling* and *adaptive modelling*. As with LEAD, adaptive sampling aims to resolve uncertainties by directing the data acquisition accordingly. Adaptive modelling aims to revise the model as a result of data acquisition.

2.2 Artificial systems: Autonomic computing

DDDAS and symbiotic simulation can be used in the design and management of artificial systems. We will look particularly at autonomic computing, which is about the automation of system and network administration. In IBM's description [31], autonomic computing "allows people to focus on the big picture because the low level tasks can be 'taught' to monitor and manage themselves". Autonomic computing is divided into four main areas *self-protection*, *self-healing*, *self-configuration* and *self-optimisation*. Software agents (autonomic managers) are given responsibility for different parts of the system (e.g. web service, Grid Middleware).

Simulation can be used by an autonomic agent to model the operation of a network and to assist the system administrator. The DDDAS process of continual absorption of data, selecting the most relevant data sources and revising the model can also be applied in this domain (although to our knowledge this has not yet been done).

Since the purpose is to allow the user to focus on the "big picture", some low-level simulations used by an autonomic agent could be purely internal and for the purpose of adaptation only (as in Figure 1). The architecture could begin in the form of an interactive system (Figure 2). As it becomes more trusted, the adaptation process can become increasingly automated and the architecture begins to look like an autonomous agent. The system administrator only needs to be concerned with the high level issues (e.g. unusual problems that affect business requirements) where a human in the loop is still required.

2.2.1 Subclass: Intrusion defence

In the "self-protection" domain of autonomic computing, simulation can be used to predict the effects of suspected intrusions on the network as well as the effects of potential responses to intrusions. This is an area that normally requires human participation, since responding in the wrong way could have serious consequences. A DDDAS agent can act as a decision-support system for a system administrator by identifying potential intrusions and predicting their effects in the current circumstances or in "what-if" scenarios. Recommendations for responses could be given, based on further simulations, showing the expected consequences of responses. Some responses may be fully automated. (The DDDAS agent can be interacting closely with the lower level autonomic manager agents, whose task is to intervene autonomously in the system being protected).

Some network modelling has already been done in the field of intrusion detection. For example, [14] simulates worm propagation in a network using a stochastic epidemic model. Agent-based simulation of a network is also possible. An agent can be any "subject" with a permissible list of actions as defined in a security policy. This can be a user or any software acting on the user's behalf (e.g. servers, applications).

In other areas of autonomic computing, DDDAS could be used to predict the effects of planned configuration changes (as required by the self-configuration component of autonomic computing).

2.3 Social Systems: Public policy decision support

It is not expected that a simulation of a social system can be "symbiotic" in the same way as for an artificial system. Instead we will consider the simulated social system as a natural system that may be modified only in the longer term by a policy change.

Public policy decision support can be regarded as a complex intelligent control system with humans in the loop. It belongs to a more general class of domains in which a limited set of resources has to be managed fairly. Specifically, we assume that (a) a set of minimal requirements have to be met (e.g. relating to environment, health, crime-prevention) and (b) competing interests and desires of participating agents should be maximally satisfied (with necessity for compromise when conflicts exist).

At any given time, these requirements will not usually be met. A major challenge is to identify which currently unsatisfied need will lead to dangerous situations in the future (e.g. social exclu-

sion leading to increased crime or ethnic tensions). This should help to prioritise the kind of policy decisions required. The future impacts of possible interventions can then be evaluated in the same way.

The general class of resource management also includes domains that are not “social policy” but still involve situations in which diverse needs and wishes must be reconciled. An example would be the management of e-science resources to support a collaboration between scientists and users with different concerns and interests (which may also be competing).

In contrast to collaboration scenarios, however, public policy has the characteristics of a safety-critical system, since the consequence of error can be life-threatening (or at least threatening to health and safety). It is useful to compare policy decision support with control of a physical system in which humans are “in the loop” and are also “stakeholders” in the protected system (e.g. a life-support system or air-traffic control). The main difference between such domains and public policy is that the timescales are much shorter.

2.3.1 Value of DDDAS to public policy

There is much existing work on agent simulations for geographical decision support systems [4, 3]. In particular, there are several applications in urban planning [9, 6, 2]. We need to ask how DDDAS can fit into such systems and what its added value is.

If a simulation is to be a reliable guide to policy making, it must be accurate in its representation of those aspects of reality that are relevant to the policy-maker’s goals. A non-DDDAS simulation is validated by comparing its results offline with empirical data. For example, a simulation could be based initially on a theory of agent behaviour (e.g. housing market reaction to environmental threats such as flood risk) and this could also be consistent with empirical observations. However, it may not accurately predict the effects of some policies due to oversimplification of the observed system. Furthermore, empirical observations carried out offline are selected in a pre-determined way and could overlook important data sources that were believed to be insignificant at the time.

The use of DDDAS may overcome some of these problems because it allows unexpected features in the observed system to affect the simulation directly and possibly contribute to a revision of the theory that might not have been discovered otherwise. Moreover, the selection of data for assimilation is not restricted by assumptions about what is important, as would be the case for “offline” validation.

We can conclude that DDDAS has the potential to provide the following advantages:

1. Reliability: the resulting model that emerges from the DDDAS process can be more reliable than one that is static and only revised manually “offline”.
2. Knowledge discovery and creativity assistance: human reasoning about social systems is often biased by one’s own experience, interests and values etc. If the model-building and revision process is partially automated, there is a possibility of discovering patterns and relationships in the data that would not be found otherwise. Similarly, the process can assist creativity in solving problems.

It is of-course not certain that these advantages can be provided in practice. We now discuss the technical challenges.

3 Technical Challenges for Social Systems

We envisage an agent-based simulation in which the DDDAS software agent uses data-driven simulation to acquire an accurate model of the world it is helping to make decisions on. The world contains active “agents” and passive objects, along with their complex relationships and interdependencies.

Note that two kinds of “agent” exist: (a) the software agent(s) which manages the simulation and (b) the simulated agents, which represent real actors in the observed system. In some scenarios

involving a software “world” this may include the simulation management agents themselves (self-monitoring) as well as other “agents” which may be artificial or human. In a social system, examples include individuals, groups, organisations, consumers, policy-makers, transport users, companies etc.

Development of a reliable decision support system requires an initial simulation, based on a model of the observed system’s behaviour (the user’s model and simulation of Figure 2). The goals and priorities of the policy process must also be specified. These are the requirements that the system ought to satisfy in an ideal situation. They could be included as part of the agent’s model M in Figure 2 and would not normally be subject to revision except through interactive consultation in the event of conflicting requirements.

DDDAS can be used to increase the accuracy of the simulation’s predictions about the observed system through adaptation. The model on which the simulation is built should be gradually adjusted until there are no more significant discrepancies or there is no further improvement in prediction accuracy. This can be done as an interactive process as discussed in Section 1.2.2 and Figure 2.

Once the simulation’s predictions have become sufficiently accurate (depending on risk assessments) it could then be used reliably to predict problems and to generate “what-if” scenarios to support decision-making.

The following technical problems are particularly challenging for social systems (although they may apply to other domains as well):

1. feasibility of assimilating social data into a simulation, considering time lag in the data collection, lack of directly controlled sensors etc.
2. assimilation of qualitative and subjective data for cognitively rich agents (involving beliefs and emotions such as hope, fear etc);
3. how to select the relevant data for assimilation using the goals and priorities of the decision support scenario;
4. how to introduce model revision and adaptation in a way that can challenge fundamental assumptions if necessary;
5. how to provide fault-tolerance in the reasoning process.

3.1 Data acquisition and assimilation

We assume that social simulation can be placed approximately in one of the following categories (but see qualifying remarks below):

(a) The simulation represents the evolution of a single observed system (e.g. a particular city, a supermarket outlet). The data collected from the observed system can fit directly into the simulation states (e.g. consumer behaviour in the simulation can be checked against consumer behaviour in the observed system directly). The two systems could run in parallel (as may be possible in a transport network) or the simulation could be adjusted as historical data becomes available. This means that the simulation can “lag behind” the actual system, but can still be useful.

(b) The simulation is of a “typical” system, not one particular instance. Its predictions are applicable to a whole class of observed systems (e.g. a typical UK city within a given population range). This is more common in agent-based simulations but more difficult to absorb data into. The ontology (objects and relations) and the states of the simulation will probably be more abstract and summarised than that of (a). Data from many different real systems are available but they must be fused and summarised into the more abstract concepts used by the simulation. General patterns (e.g. in the form of rules) may be mined from the data and they will have to be checked for consistency with the simulation predictions.

Type (a) is the normal architecture for DDDAS simulations. Typically the simulation and the real system run in parallel so that the data are collected in real-time. Probably this is not realistic for most social science scenarios. Instead, we expect to choose a simulation architecture that is somewhere on a spectrum between (a) and (b). It is very unlikely to be purely (a) as we do not need data on every single instance of an agent in reality (there are also ethical questions with this). Instead of monitoring and simulating every individual, the simulated agents should represent anonymous persons or groups that play a particular *role* (e.g. the house buyers in the city).

Since this involves some degree of generalisation (e.g. a “typical” first time house buyer), the data to be absorbed has to be summarised so that it says something about the “typical” groups of individuals. Data mining tools are important for the discovery of patterns in the data that may be used to update or revise models.

If the agents have their own internal beliefs and motivations then some of the assimilated data could be used to update these beliefs. Potential data sources may include “speech acts” (e.g. newspaper reports) which would require textual analysis. In particular, data involving positive or negative *evaluation* of the agent’s own state and its environment may be important for predicting potential social problems [1].

3.2 Selection of data for absorption

The states predicted by the simulation could be used to select the kind of data to be absorbed. Whether the simulation is of type (a) or (b), masses of data can be collected, and it is important to focus on the most relevant and to ask the right questions about the real system. The following possibilities exist:

- Evaluation-directed selection: If a predicted event is substantially positive (e.g. lower crime rates) or negative (e.g. significant violence) the data selection can be focused on the context of the positive or negative event (e.g. what kinds of behaviours have preceded the event and do they tend to be associated with such events in reality? Database query and data mining tools are required here. If the evidence does not support the predictions, the simulation may be flawed and could be subjected to revision. If correct, the data mining tools have been focused in a way that might not have happened otherwise and the simulation has pointed to some valuable knowledge hidden in the masses of data.
- Anomaly-directed selection: If there is a significant discrepancy (anomaly) between a predicted event and the reality, the features of the data that are unexpected could be used to formulate new data queries. This is similar to the uncertainty driven selection of [20] and [18] described earlier.

In this way, the simulation draws attention to aspects of the real system that may not have been known otherwise. This could in turn be used to implement changes (new policies).

3.3 Model revision

Model revision (or belief revision) is a well-known problem in AI. Many AI systems use symbolic logic to derive new statements about the world from current ones. Non-monotonic reasoning involves making deductions using tentative assumptions and when necessary making subsequent revisions to these assumptions as new information becomes available (See e.g. [22], Ch. 7).

A non-monotonic system could play a role in model revision in a DDDAS process. On its own, however, it would have serious limitations. Formal symbolic reasoning is governed by a pre-defined ontology (entities and relations) and this can lead to brittleness in unforeseen situations for which the ontology was not designed. Reasoning about social systems is particularly vulnerable because the ontology will only represent a very restricted part of the real system and will be designed according to its expressiveness and meaningfulness to a human expert. Most of this meaningfulness is implicit and only affects the human’s decision-making, since it have been acquired by human experience in the world. (This ontology is the basis of U and its translation into M in Figure 2).

The AI system treats the symbols of the ontology as formal patterns only, which are transformed into other patterns according to rules. It does not have an independent “experience” of them. Therefore any belief revision may not take into account important features of the underlying system for which the ontology is not suited.

This problem has been identified as “symbol grounding” [13]. The problem also exists in simulations [10], since they are also dependent on an ontology which may have meaning only for an external interpreter. We use the broader term “semantic grounding”, to refer to the checking of the validity of a model by independently interacting with the world and adapting to it autonomously.

It could be argued that “grounding” is fundamentally about the learning and refining of concepts by physically interacting with the world and letting the sensory impressions affect the concept formation. Interaction does not have to be physical, however. The important issue is the “data-driven” nature of the concept formation. This is clearly relevant to a DDDAS architecture. However, we must first identify how much of the “semantic grounding” problem we actually need to solve in order to have a sufficiently robust and adaptive system for decision support.

3.4 Semantic grounding and fault-tolerance

Lack of semantic grounding can be regarded as a general fault-tolerance problem (it is not just applicable to model revision). The questions we wish to ask are:

- What are the dangers of relying on a non-grounded symbol system when making decisions on social issues?
- What kind of architecture could overcome these problems? In other words, how do we introduce fault-tolerance?

The first danger relates to the reliance on a single ontology (a single point of failure) which is effectively just one viewpoint of a social system. Furthermore, a non-grounded system is like relying on “hearsay” without having the opportunity to check its validity by experiencing the situation directly and independently testing it.

Architectural features to overcome these problems include the following:

1. Avoid a single point of failure by adding diverse ontologies and reasoning systems (involving multiple software agents).
2. Ensure that each agent interacts independently with its environment to associate symbolic concepts with additional features discovered as a result of the interaction. This becomes the agent’s independent “experience” of the concept which is not included explicitly in the ontology.

3.4.1 Multiple ontologies

There are two ways in which multiple ontologies can co-exist:

1. They can be different viewpoints or descriptions of the observed social system. For example, one description might involve simple agents with objectively determined states and actions (e.g. buying, selling, mobility etc.) using mostly numerical modelling; another might emphasise the beliefs and emotional states of agents as the primary cause of their decisions and actions.
2. What begins as a single model could be defined on multiple levels of abstraction. For example, the lowest level might define a human agent as a very specific kind of individual and describe the detailed stages of commuting to work, going shopping, buying particular products etc. This could be on the level of “raw events” that could be sensed in real-time or retrieved from databases. In other words, very little data fusion and summarisation would be necessary. The highest level might describe sections of the population or classes of organisation as abstract “agents” and define their influences on each other and their goals.

If we return to Figure 2, multiple versions of M could be defined. For (1) above, there may be several versions of U which translate to corresponding versions of M , or only one version of U translates to multiple agent models (e.g. using different representations). It is reasonable to have separate software agents, each with its own model. For (2) one agent has multiple versions of M on different levels of abstraction. Both of these approaches could be combined. Figure 3 shows an example configuration

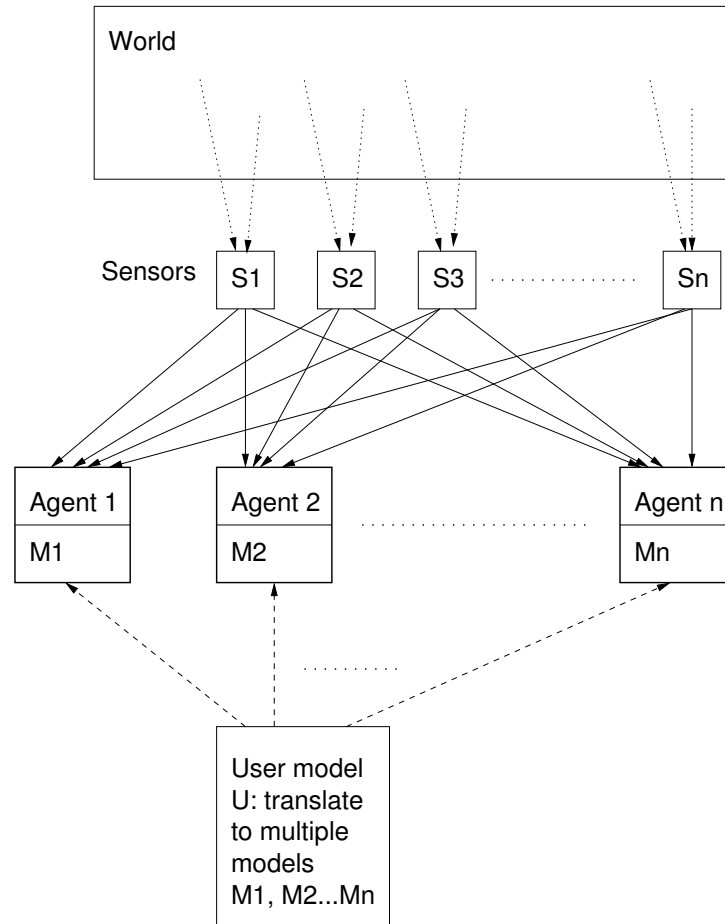


Figure 3: Multiple agents, each with its own model of the world

of (1), where a single user defined model is translated into different agent models M_1, M_2, \dots, M_n . Note also that the requirements (goals) of the policy process would also have to undergo the same translations.

We can imagine each agent in the figure also having a hierarchy of models, each producing its own internal simulation as necessary. For example, M_1 could be further subdivided into $M_{11}, M_{12}, \dots, M_{1m}$, where M_{11} is the most detailed and concrete and could merely predict the next set of sensor measurements. On the other extreme, M_{1m} could predict the general trend in the behaviour of a particular class of agents.

The sensors in the diagram are shown as external shared components (not “owned” by each agent). This is more likely to happen in realistic distributed scenarios (e.g. environmental sensing).

If there is a human in the loop, a significant disagreement between agents (with different models) may not be a serious problem but could indicate that more detailed information is required (as discussed in Section 3.2).

3.4.2 Independent adaptation to the environment

Architectures already exist in AI which could help to connect symbolic models with independent experience of the environment. For example, a hybrid agent architecture such as that in e.g. [29, 26]

may be appropriate. A hybrid architecture is one that integrates the symbolic tradition of AI with newer behaviour-based approaches introduced by Brookes [5]. In such a system, high level deliberative reasoning is coupled with a more data-driven reactive layer whose actions are *not* necessarily restricted by the pre-defined ontology governing the reasoning but may partly override it. “Behaviour-based” approaches to adaptation include reinforcement learning and neural networks. See [16] for a summary). They are “data-driven” because the result of the learning process is determined largely by low-level features in the environment and less by any pre-defined knowledge in an ontology.

To show how a hybrid architecture can contribute to a DDDAS system, we first look at the data assimilation process. The low-level sensory patterns must be summarised and translated into concepts in the top level ontology (in AI called a “perception hierarchy”). This is particularly necessary for simulations of Type (b) in Section 3.1. This is also where a hierarchy of ontologies (and model versions) could exist on different levels of abstraction (see also Figure 3). A hybrid agent architecture may be applicable to each agent in Figure 3, but ideally they would need their own sensors and effectors (not shown in the diagram) to explore and interact with the world. (We will discuss variants of this later, however).

Simulations may also exist on lower levels of the hierarchy [27, 12], but need only serve the purpose of the agent’s adaptation (they are not necessarily interesting for the user but could just have an “autonomic” function as defined in Sections 1.2.1 and 2.2).

3.4.3 Anomaly-directed model revision

There are many possible ways in which a model revision process may be triggered. For example, there may be inconsistencies between expected relationships in the data (as stated by the model) and the apparent actual relationships. If such anomalies are detected, the agent can focus on the data that appears to be causing the problem and “explore” it (e.g. sampling the data in different ways with large numbers of database queries). Such exploration can be similar to a robot exploring its environment. Autonomous adaptation results if the model governing the simulation is adjusted as a result of such exploration. Concepts in the ontology are “grounded” if they are associated with observed effects that were experienced during exploration and these effects are consistent with model descriptions of them.

Formation of new concepts is a more radical form of adaptation and means that completely new entities and relations are generated (thus revising the ontology on which the model is based). Concept formation may be necessary if the new data does not “fit” easily into an existing concept because a situation has arisen that was not anticipated by the designer. Detection of such anomalies can trigger more intensive exploration of the relevant “spaces” (whether physical or information-related) as discussed in Section 3.2. Recent work in concept formation involving symbol grounding includes [23, 17, 11].

An ongoing research question is the degree to which the data-driven layer of a hybrid architecture can “interrupt” or influence the high level reasoning (e.g. if a dangerous situation is detected). Similarly, the degree to which the learning process is data-driven or top-down is important. For DDDAS, this translates into the following research questions: how much should the model (and the simulation states) control the selection of data for assimilation, and conversely what is the boundary between absorption of anomalous data into existing concepts and the creation of new concepts?

4 Interaction with the World

The agent architecture mentioned in the previous section assumes the ability to interact with its environment and to learn autonomously. However, in most social science scenarios the interaction is expected to require human intermediaries and to involve indirect access via speech acts, meaning that “grounding” becomes more difficult. To understand this in detail, we will consider three different categories of access to the world: physical, social (speech acts) and online activities.

4.1 Direct access to physical environment

From an AI point of view, it is simplest to begin with the situation where the DDDAS process is being run by a robotic agent which is situated in the physical world it is helping to model. I.e. the relation to the environment is as shown in Figure 1, although the modelling relations may be more like Figure 2. The robot's interaction with the world via sensors and effectors acts as the data source for its ongoing simulation. In addition to the physical characteristics of the environment, the simulation may include the robot itself and possibly other robots it is cooperating with. Examples include remote vehicles (e.g. Mars exploration or underwater monitoring).

In reality it is unlikely that a DDDAS system would take this simplified form. Instead the computationally intensive aspects could be provided by distributed computing services such as the Grid, while the data collection aspects could be done by mobile robots or "dumb" sensors. Any direct action required on the world (e.g. as part of a symbiotic simulation) could also be done by directing the robots. In this case the robots have less autonomy since they are part of wider distributed network. They could still have their own behaviour-based adaptation capability but this would be overridden by high level directives if necessary. Conversely, there may be "alarm" situations where the low-level robots could alert or even interrupt the higher levels.

If we return to Figure 3, one possible scenario is that "sensors" in this figure could be semi-autonomous robots, while the agents could act on behalf of different users of the system, who have their own models of the world (multiple versions of U). Specialist agents are then needed to allocate resources correctly according to current priorities.

The robots themselves may be included as simulated "agents" in the world being modelled if, for example, scientists are interested in the effects of the environment on the reliability of the robots or on their ability to communicate among themselves.

4.2 Indirect access: speech acts

Speech acts of other agents are a more indirect way of accessing the world than the use of sensors. However, the distinction between the two is not sharp, since many data sources that could be called "sensors" are actually "event detectors" (i.e. they "say" that an event has occurred). In general, access becomes more indirect the more it relies on information that is pre-processed by other agents that are themselves autonomous. Relying on speech acts by humans is just an extreme example.

Many social science datasets involve speech acts - e.g. results of questionnaires asking whether people have moved house recently and whether they are satisfied with their current housing situation.

On the other end of the spectrum, a sensor that sends raw data and can be controlled by an agent provides the most direct access to the world. This is where the symbols used by the agent can have the most "grounding". Clearly, software agents do not have such direct access to the social world, as they are not situated in this world (although some advances could be made here eventually).

4.3 Direct access: online activities

In addition to observing the human agents' interactions among themselves by using indirect information sources (e.g. reports on economic activities), the agents' interactions with various online services can also be observed and modelled. This may be done by observing the various network usage activities and developing agent models based on general patterns in these activities. The resulting model may then be cross-checked for consistency with the more abstract social models. As with physical interaction, this can also provide a form of "grounding" for the high level agent models.

Activities of agents in the modelled social system can overlap with their activities as competitors for computing or information resources. For example, the infrastructure of the network may be attacked as part of the social interactions being modelled. If there are frequent security problems, this could provide supporting evidence for potential conflicts in the observed social system.

It is likely that a social science DDDAS agent will need to consult a specialist source to obtain the relevant information (e.g. on network activities). As mentioned in Section 2.2, the underlying

network and middleware may be managed by “autonomic agents” with the responsibility to manage competing resources and to protect the network from intrusions (among others). They may even have their own DDDAS systems to assist system administrators. If such autonomic agents exist, they can be “points of contact” for information on usage of network services and Grid resources. The data provided by the autonomic agents could be in a summarised form, as the low-level details of network activity need not be part of the ontology used to model the social system.

Conversely an intrusion defence agent may query the social science agent for information on the social and legal context of an intrusion. The actions of modelled agents in the intrusion defence system may be connected with agents that are modelled as part of the social system (for example if an intrusion is associated with a crime in progress).

If we return again to Figure 3, the “world” could be divided into various sections, each representing a “space” that an agent can explore and interact with using specialist sensor and effectors. The space may in some cases be physical or geographical; in other cases it will be an information world.

5 Summary and Conclusions

DDDAS has the potential to improve the reliability of agent-based simulations used for decision support systems and can also assist with knowledge discovery and creativity in the social science domain. Due to the non-real-time nature of most social science data, it is more likely that the simulation is based on a typical system rather than a specific one that is being concurrently monitored. This means that models will tend to be abstract and that the data assimilation will depend considerably on data mining and fusion from multiple sources over a longer timescale.

To reduce the brittleness associated with “semantically ungrounded” concepts in social simulations, we can conclude that the following is important:

1. *Multiple ontologies* representing alternative descriptions of the world are advantageous. They can be the basis for different models, which can generate their own simulations. Potential problems can be detected if there are significant disagreements between model predictions.
2. *Autonomous learning and adaptation* is required, involving independent interaction with the world (i.e. data sources) in order to check the validity of a model and to revise it as necessary. There are different “spaces” that an agent can explore in order to adapt. We have outlined three types: physical, social data sources and online activities.
3. *Cooperation between heterogeneous agents* acting in different domains and levels of abstraction is important in order to exploit diverse sources of information that can be connected together. This increases the fault-tolerance of the system. The DDDAS agent itself is a service that is embedded in a wider Grid infrastructure and should exchange information with agents managing other services. (For example, an agent managing the Grid middleware may have useful information on resource usage which could be coupled with competing interests in the social system being modelled).

Acknowledgements

This research was supported by the Paul and Yaunbi Ramsey Fund.

References

- [1] Khurshid Ahmad, Lee Gillam, and David Cheng. Textual and Quantitative Analysis: Towards a new, e-mediated Social Science. In *First International Conference on e-Social Science*, Manchester, UK, June 2005.

- [2] Arnaud Banos, Thomas Thevenin, Sonia Chardonnel, Christophe Lang, and Nicolas Marilleau. Simulating the Swarming City: a MAS Approach. In *Computers in Urban Planning and Urban Management (CUPUM'05)*, London, UK, July 2005.
- [3] Mark Birkin, Haibo Chen, Martin Clarke, Justin Keen, Phil Rees, and Jie Xu. MOSES: Modelling and Simulation for e-Social Science. In *First International Conference on e-Social Science*, Manchester, UK, June 2005.
- [4] Mark Birkin, Pete Dew, Olga Macfarland, and John Hodrien. HYDRA: A prototype grid-enabled spatial decision support system. In *First International Conference on e-Social Science*, Manchester, UK, June 2005.
- [5] R. A. Brooks. A Robust Layered Control System For A Mobile Robot. *IEEE Journal Of Robotics And Automation*, RA-2:14–23, April 1986.
- [6] Mario Campanella, Ana Dzokic, Marc Neelen, Milica Topalovic, Ivan Kucina, and Ivan Lazarevic. Self-Organised Informal Markets: Uncontrolled Forces Shaping the City. In *Computers in Urban Planning and Urban Management (CUPUM'05)*, London, UK, July 2005.
- [7] Frederica Darema. Grid Computing and Beyond: The Context of Dynamic Data Driven Applications Systems. *Proceedings of the IEEE: Special Issue on Grid Computing*, 93(3):692–697, March 2005.
- [8] Johan de Kleer and Brian C. Williams. Diagnosing multiple faults. *Artificial Intelligence*, 32(1):97–130, 1987.
- [9] Oswald Devisch, Theo Arentze, Aloys Borgers, and Harry Timmermans. An Agent-Based Model of Residential Choice Dynamics in Imperfect, Non-Stationary Housing Markets. In *Computers in Urban Planning and Urban Management (CUPUM'05)*, London, UK, July 2005.
- [10] Bruce Edmonds and Scott Moss. From KISS to KIDS - an anti-simplistic modelling approach. In *Joint Workshop on Multi-Agent and Multi-Agent-Based Simulation (MAMABS 2004) at the 3rd Conference on Autonomous Agents and Multi-Agent Systems (AAMAS-2004)*, Columbia University, New York City, July 2004.
- [11] Peter Gorniak and Deb Roy. Grounded semantic composition for visual scenes. *Journal of Artificial Intelligence Research*, 21, 2004.
- [12] R. Grush. The emulation theory of representation: Motor control, imagery, and perception. *Behavioral and Brain Sciences*, 27:377–442, 2004.
- [13] S. Harnad. The symbol grounding problem. *Physica D*, 42:335–346, 1990.
- [14] Michael Liljenstam, David M. Nicol, Vincent H. Berk, and Robert S. Gray. Simulating realistic network worm traffic for worm warning system design and testing. In *Proceedings of the 2003 ACM workshop on Rapid Malcode (WORM)*, pages 24–33. ACM Press, 2003.
- [15] Malcolm Yoke Hean Low, Kong Wei Lye, Peter Lendermann, Stephen John Turner, Reman Tat Wee Chim, and Surya Hadisaputra Leo. An Agent-based Approach for Managing Symbiotic Simulation of Semiconductor Assembly and Test Operation. In *Fourth International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS 2005)*, Utrecht, The Netherlands, July 2005.
- [16] George F. Luger. *Artificial Intelligence, 5th Edition*. Addison Wesley, Harlow, England, 2005.
- [17] Tim Oates, Matthew D. Schmill, and Paul R. Cohen. Identifying qualitatively different outcomes of actions: Gaining autonomy through learning. In Carles Sierra, Maria Gini, and Jeffrey S. Rosenschein, editors, *Proceedings of the Fourth International Conference on Autonomous Agents*, pages 110–111, Barcelona, Catalonia, Spain, 2000. ACM Press.

- [18] N.M. Patrikalakis, J.J. McCarthy, A.R. Robinson, H. Schmidt, C. Evangelinos, P.J. Haley, S. Lalis, P.F.J. Lermustaux, R. Tian, W.G. Leslie, and W. Cho. Towards a dynamic data driven system for rapid adaptive interdisciplinary ocean forecasting. In F. Dorema, editor, *Dynamic Data-Driven Application Systems*. Kluwer Academic Publishers, Amsterdam, 2004.
- [19] Barney Pell, Douglas E. Bernard, Steve A. Chien, Erann Gat, Nicola Muscettola, P. Pandurang Nayak, Michael D. Wagner, and Brian C. Williams. An autonomous spacecraft agent prototype. In W. Lewis Johnson and Barbara Hayes-Roth, editors, *Proceedings of the First International Conference on Autonomous Agents (Agents'97)*, pages 253–261, New York, 1997. ACM Press.
- [20] Beth Plale, Dennis Gannon, Dan Reed, Sara Graves, Kelvin Droegemeier, Bob Wilhelmson, and Mohan Ramamurthy. Towards Dynamically Adaptive Weather Analysis and Forecasting in LEAD. In *Workshop on Dynamic data Driven Application Systems at the International Conference on Computational Science (ICCS 2005)*, Atlanta, USA, May 2005.
- [21] Marc D. Rayman, Philip Varghese, David H. Lehman, and Leslie L. Livesay. Results From The Deep Space 1 Technology Validation Mission. *Acta Astronautica*, 47:475–488, 2000.
- [22] Elaine Rich and Kevin Knight. *Artificial Intelligence*. McGraw-Hill Higher Education, 1990.
- [23] Deb Roy and Alex Pentland. Learning words from sights and sounds: A computational model. *Cognitive Science*, 26(1):113–146, 2002.
- [24] Murray Shanahan. Consciousness, emotion, and imagination: A brain-inspired architecture for cognitive robotics. In *Proceedings of the AISB 2005 Workshop: Next Generation Approaches to Machine Consciousness*, pages 26–35, 2005.
- [25] A. Sloman. Beyond shallow models of emotion. *Cognitive Processing: International Quarterly of Cognitive Science*, 2(1):177–198, 2001.
- [26] A. Sloman and M. Scheutz. A Framework for Comparing Agent Architectures. In *Proceedings of UKCI'02, UK Workshop on Computational Intelligence*, Birmingham, UK, 2002.
- [27] Aaron Sloman. Cognitive Systems for Cognitive Assistants (COSY) Deliverable Report 2.1: Requirements Study for Representations. Technical Report COSY-TR-0507, School of Computer Science, University of Birmingham, 2005.
- [28] Aaron Sloman and Jackie Chappell. The Altricial Precocial Spectrum for Robots. In *Proceedings of IJCAI'05*, Edinburgh, UK, August 2005.
- [29] Aaron Sloman and Brian Logan. Architectures and tools for human-like agents, 1998.
- [30] Aaron Sloman and Brian Logan. Building Cognitively Rich Agents using the SIM-AGENT Toolkit. *Communications of the Association of Computing Machinery (CACM)*, 43(2):71–77, 1999.
- [31] Daniel H. Steinberg. What you need to know now about autonomic computing, Part 1: Introduction and Overview <http://www-106.ibm.com/developerworks/ibm/library/i-autonom1/> last consulted February 2005.