A Minimal Prototype for Integrating GIS and Geographic Simulation through Geographic Automata Systems♦

Itzhak Benenson[1], Paul M. Torrens[2]

[1]Environment Simulation Laboratory Porter School of Environmental Studies, and Department of Geography and Human Environment Tel Aviv University, Israel, [2]Department of Geography, University of Utah, United States. {bennya@post.tau.ac.il, torrens@geosimulation.com}

Abstract

A new approach is introduced, whereby patently spatial simulation methodology is applied to simulating discrete, dynamic, and action-oriented spatial systems, combining cellular automata and multi-agent systems in a spatial context. In this paper, we propose a minimal prototype for integrating GIS and geosimulation into what we term *Geographic Automata Systems* (GAS), the latter sufficient for the formalization of the majority of abstract and real-world high-resolution models. A software environment for urban simulation, which directly implements GAS formalism, is briefly described and an example of its application is provided.

1. Introduction

New forms of simulation have come into popular use in geography and social science in recent years, supported by an array of advances both in the geographical sciences and in fields outside geography. These models and simulations can be characterized by a distinctly innovative approach to modeling—the geosimulation approach. Geosimulation is concerned with automata-based methodologies for simulating collectives of discrete, dynamic, and action-oriented spatial objects, combining cellular automata and multi-agent systems in a spatial context (Benenson and Torrens 2004; Benenson and Torrens 2004). In geosimulation-style models, urban phenomena as a whole are considered as the outcome of the collective dynamics of multiple animate and inanimate urban objects.

Geosimulation models are more commonly based on cellular automata (CA) and multi-agent systems (MAS). Applied in isolation, CA and MAS approaches have been used to simulate a wide variety of urban phenomena and there is a natural imperative to combine these frameworks for exploratory and applied simulation in urban geography (Torrens 2004, 2003, 2002). Nevertheless, the direct amalgamation of CA and MAS eventually suffers from awkward compromises, a function of the necessity for CA partition of urban space into cells. Given cell partition, no matter whether units are identical or vary in size and shape, cells are either granted some degree of 'agency' and are simply reinterpreted as artificial agents (Box 2001) or MAS are imposed on top of CA and

♦ Benenson, I. & Torrens, P.M. (2004) "A minimal prototype for integrating GIS and geographic simulation through Geographic Automata Systems". In *GeoDynamics*, P. Atkinson, G. Foody, S. Darby, F. Wu (Eds.) Florida: CRC Press, pp. 347-369

simulated agents respond to averaged cell conditions (Benenson 1998; Polhill, Gotts, and Law 2001). These frameworks are certainly useful, especially in studying abstract models, but in a real-world milieu are a function of the limitations of the available tools rather than the structure of real urban systems or the laws of their behavior (see Agarwal, this volume, for a discussion of the limitations of agent-based models).

The dataware for the real-world simulations is usually provided by Geographic Information Systems (GIS), which have an integral role in the development of geosimulation models. Dramatic changes in geographic databases during the last decades of the Twentieth Century have ushered in a new wave of urban modeling (Benenson, Omer, and Hatna 2002; Benenson and Omer 2003). Automated procedures for data collection—remote sensing by spectrometer, aerial photography, scanners, video, etc.—have provided new information sources at fine resolutions, both spatial and temporal (Torrens 2004). New methodologies for manipulating and interpreting spatial data developed by geographic information science and implemented in GIS have created added-value for these data. Information collection is much more pervasive than before (Brown and Duguid 2000), and high-resolution databases for urban land-use, population, real estate, and transport are now more widespread (Benenson and Omer 2003). Cellular spaces populated with agents are too limiting an interface between GIS and geosimulation, and there remains much potential for fusing GIS and geosimulation into full-blown, symbiotic simulation systems. In this paper, we propose a minimal but sufficient framework for integrating GIS, CA, and MAS into what we term *Geographic Automata Systems*. This approach is intended to be 'down to earth', and we will demonstrate the implementation of the approach, in this chapter, with reference to GAS-based software developed at the Environment Simulation Laboratory of the Porter School of Environmental Studies at the University of Tel Aviv—the *Object-Based Environment for Urban Simulations* (OBEUS). This chapter builds on discussions of urban geocomputation (Torrens and O'Sullivan 2000; Benenson and Omer 2000) and software environments for geocomputational research in urban contexts (Benenson, Aronovich, and Noam 2001), which were partially presented at previous Geocomputation meetings.

2. The basic automata framework

Put very simply, an automaton is a processing mechanism; a discrete entity, which has some form of input and internal states. It changes states over time according to a set of rules that take information from an automaton's own state and various inputs from outside the automaton to determine a new state in a subsequent time step. Formally, an automaton, **A**, can be represented by means of a set of *states* **S** and a set of *transition rules* **T**.

$$\mathbf{A} \sim (\mathbf{S}, \mathbf{T}) \qquad (1)$$

Transition rules define an automaton's state, $S_{t+1,}$ at time step t + 1 depending on its state, S_t (S_t, $S_{t+1} \in S$), and *input*, I_t, at time step t:

$$\mathbf{T}: (S_t, I_t) \rightarrow S_{t+1} \qquad (2)$$

Automata are discrete with respect to time and have the ability to change according to predetermined rules based on internal (**S**) and external (**I**) information.

In terms of urban applications, automata lend themselves to specification as city simulations with myriad states and transition rules. However, to make sense, an individual automaton should be as simple as possible in terms of states, transition rules, and internal information (Torrens and O'Sullivan 2001). Simplicity is a characteristic of the most popular automata tools in urban geography, Cellular Automata —a system of spatially located and inter-connected automata.

2.1. From general to cellular automata

CA are arrangements of individual automata in a partitioned space, where each unit (cell) is considered as an automaton **A**, for which input information **I** necessary for the application of transition rules **T** is drawn from **A**'s *neighborhood* **R**. In urban applications, cells are most commonly used to represent land units with state representing possible land-uses (White, Engelen, and Uljee 1997). Usually, CA lattices are partitioned as a regular square or hexagonal grid. We can specify (1) – (2) to introduce an automaton, **A**, belonging to a CA lattice as follows:

$$\mathbf{A} \sim (\mathbf{S}, \mathbf{T}, \mathbf{R}) \qquad (3)$$

where **R** denotes automata neighboring **A**.

Although there are direct analogies between land parcels and cells on the one hand and land-uses and cell states on the other, there were no geographical applications of CA models before the 1990s. There were a few examples published in the 1970s (Albin 1975; Chapin and Weiss 1968; Nakajima 1977; Tobler 1970, 1979), but the field was largely ignored in terms of research until interest was revived in the 1980s (Phipps 1989; Couclelis 1985). Beginning in the 1990s, CA modeling became a popular research activity in geography, with pioneering applications in urban geography (Batty, Couclelis, and Eichen 1997; O'Sullivan and Torrens 2000).

In terms of space, neighborhood relationships are important for rendering CA as *spatial* systems. In basic CA, neighborhoods have identical form for each automaton, e.g., Moore or von Neumann (Figure 1a). In the last decade it became clear, however, that reliance on regular partitions of space is largely superficial in urban contexts (Torrens and O'Sullivan 2001). Consequently, CA have been implemented on irregular networks (Figure 1b), or partitions given by GIS-based coverage of land parcels or Voronoi tessellations (Figure 1c) (Shi and Pang 2000; Semboloni 2000; O'Sullivan 2001; Benenson, Omer, and Hatna 2002). An assortment of definitions of neighborhoods, based on connectivity, adjacency, or distance can be applied to these generalized CA, where the form of the neighborhood and the number of neighbors varies between automata.

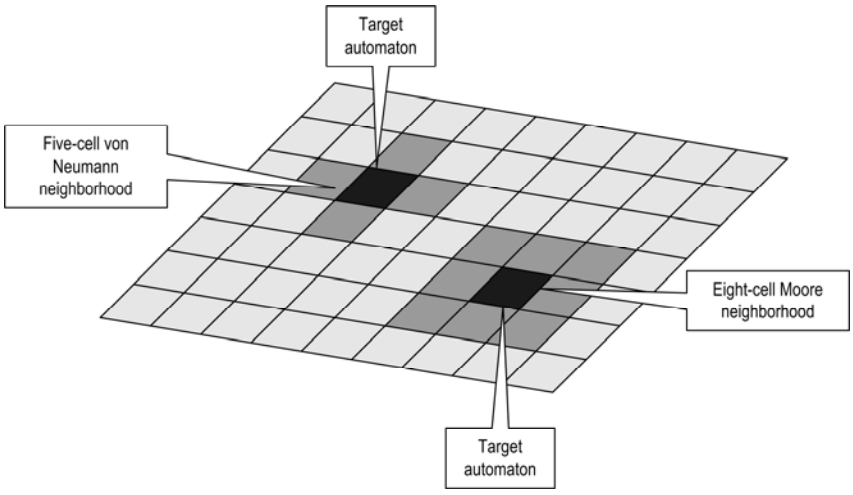

Figure 1a. Automata defined on a two-dimensional regular lattice, with von Neumann and Moore neighborhoods represented.

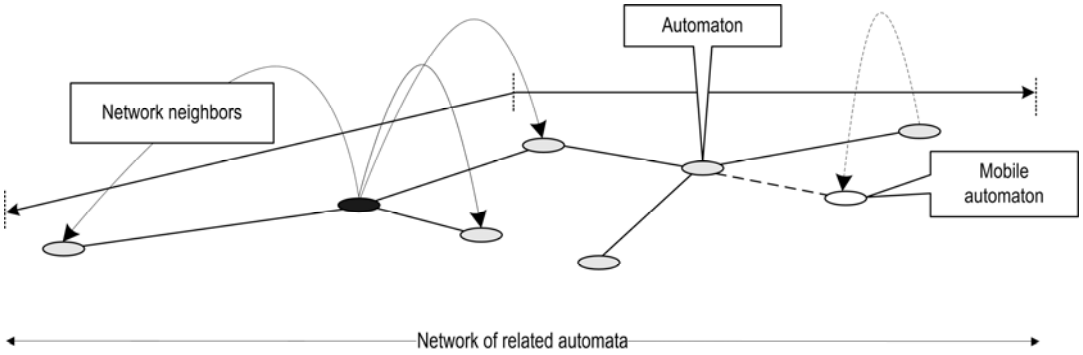

Figure 1b. Automata defined on a two-dimensional network.

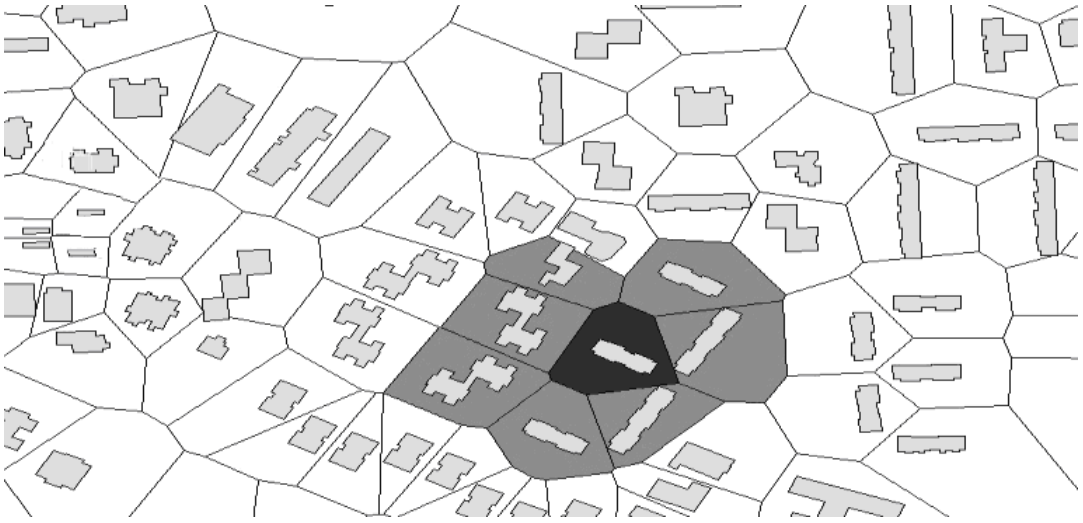

Figure 1c. Automata defined on a Voronoi partition of two-dimensional urban space, based on property coverage.

There is no conceptual difference between irregular and regular CA; yet, an inherent weakness of the CA is the inability of automata cells to move within the lattice in which they reside. Despite repeated attempts to mimic units' mobility (Portugali, Benenson, and Omer 1994; Schofisch and Hadeler 1996; Wahle et al. 2001), the genuine inability to allow for automata movement in the CA framework catalyzed geographers' recent interest in MAS. This tendency is especially strong in urban geography, where the CA framework is regarded as insufficient in dealing with mobile objects such as pedestrians (Torrens 2004), migrating households, or relocating firms.

2.2. From general to agent automata

Fundamentally, agents are automata and, thus, incorporate all of the features of basic automata that have just been discussed. However, there are some important distinctions between general and agent automata, largely owing to the fact that the latter are generally interpreted for representation of autonomous *decision-makers* (Kohler 2000; Epstein 1999). In urban studies, the states **S** of agent automata in MAS are usually designed to represent socioeconomic characteristics; transition rules are formulated as rules of agent decision-making, and correspond to human-like *behaviors*.

In the social sciences outside geography, work in agent-based simulation is usually *non-spatial*; many of the decisions and behaviors of geographic *agents* are, however, spatial in nature. Consequently, the states **S** of *geographic* MAS should include agents' *location*, and transition rules **T** should reflect, thus, the ability of agents to relocate.

Human-based interpretations of MAS have their foundation in the work of Schelling and Sakoda (Sakoda 1971; Schelling 1969, 1971, 1974, 1978) and despite initial interest (Albin 1975) the field remained relatively quiet for two decades after that. Just as with CA, the tool began to feature prominently in the geographical literature only in the mid-1990s (Portugali, Benenson, and Omer 1997; Sanders et al. 1997; Benenson 1999; Dijkstra, Timmermans, and Jessurun 2000). Until recently, the main thrust of MAS research in geography involved populating regular CA with agents of one or several kinds, which could *diffuse* between CA cells. Often, it is assumed that agents' migration behavior depends on the properties of neighboring cells and neighbors (Portugali 2000; Epstein and Axtell 1996). Very recently, agent-based models have been designed, which locate agents in relation to real-world geographic features, such as houses or roads, the latter stored as GIS layers (Benenson, Omer, and Hatna 2002) or landscape units—pathways and view points (Gimblett 2002).

Despite the widely acknowledged suitability of automata tools for geographic modeling (Gimblett 2002), there has been relatively little exploration into addressing these limitations and developing patently *spatial* automata tools for urban simulation. In the framework described in this paper, spatial abilities are treated with paramount importance and we define a class of automata that is capable of supporting explicit expression of processes, as *geography* comprehends them. In what follows, we aim at formulating a *minimal framework* and, thus, formulate it on the basis of an agent as an autonomous mobile object that reacts to the environmental information by changing its state and location. In terms of MAS, these agents are called weak agents and reactive agents (Woolridge and Jennings 1995), and we assume that they are sufficient for the vast majority of high-resolution geographic applications.

2.3. What geography needs from automata systems

Focusing on space, we might identify three internal geographic mechanisms that are essential to an urban automata system:

- A typology of entities regarding their use of space in which they are situated;
- The spatial relationships between entities;
- The processes governing the changes of their location in space.

Simulating spatial systems, then, involves explicit formulation of these three components and neither CA nor MAS fully provide the necessary framework. The geography of the CA framework is problematic, for example, for urban simulation because CA are incapable of representing autonomously mobile entities. At the same time, MAS are weak as a single tool because of common underestimation of the importance of space and movement behavior.

It is evident, then, that there is a need for uniting CA and MAS formalisms in such a way as to directly reflect a geographic and object-based view of urban systems. The GAS framework that we propose attempts to satisfy that demand.

2.4. An idea: urban object ≡ Geographic Automaton

As a spatial science, geography concerns itself with the behavior and distribution of *objects in space*. In urban geography, these are urban agents—householders, pedestrians, vehicles—and urban features—land parcels, shops, roads, sidewalks, etc. In dynamic spatial systems, all these objects change their properties and/or location; the goal of a geographic model is to simulate these activities and their consequences, often at multiple scales.

In developing Geographic Automata Systems, our aim is to infuse spatial properties into automata tools and adopt an *automata-based* view of urban systems. Objects are conceptualized as *geographic automata*, with focus on their *spatial properties and behaviors*. Under this framework, a city system can be modeled as an ensemble of geographic automata, i.e. a Geographic Automata System.

3. Formal definition of Geographic Automata Systems

Geographic Automata Systems consist of interacting *geographic automata* of various types. In general, automata are characterized by states **S**, description of their input **I**, and transition rules **T**. In the case of geographic automata, we re-interpret and extend these components to enable the explicit consideration of *space* and *spatial behavior*.

Specification of state set **S**:

- In addition to non-spatial *states*, geographic automata are also characterized by their *locations.*
- Instead of the fixed location of CA automata, we introduce a set of *geo-referencing rules* for situating geographic automata in space.

Specification of input information **I**:

- Instead of relying on fixed neighborhood patterns that are incapable of being varied in space or time, we define *neighborhood relations* that can change in

time and determine the automata providing input information for a given automaton.
- Neighborhood relations can thus change; these changes are governed by *neighborhood rules*.

Specification of transition rules **T**:
- *State transition rules* specify the changes of non-spatial states.
- The ability to change location is provided by a set of *movement rules* that allow for navigation of geographic automata in their simulated environments.

Based on these specifications, we can formulate a minimal, yet adequate, framework for geographic modeling—or geosimulation.

Let us use **K** to denote a set of *types* of automata; following this we will feature each of the previously-mentioned properties and rules that determine the dynamics of those automata in space. Essentially, we are constructing a system of geographic automata from the bottom-up.

The automata basis of the system is captured by a set of *states* **S**, associated with the GAS, (consisting of subsets of states $\mathbf{S}^k$ of automata of each type $k \in \mathbf{K}$) and a set of *state transition rules* $\mathbf{T_S}$, used to determine how automata states should change over time.

According to general definition (1) – (2), state transitions and changes in location for geographic automata depend on automata themselves and on input (**I**), given by the states of *neighbors*. To specify the input information we use **R** to denote the neighbors of the automata and $\mathbf{N_R}$ to specify the *neighborhood transition rules* that govern how automata relate to the other automata in their vicinity.

The spatial nature of the geographic automata is encapsulated by information on automata location. Let us denote this with **L**, the *geo-referencing conventions* employed to locate automata in the system, and $\mathbf{M_L}$, the *movement rules*, governing changes in their location.

Altogether, a GAS, **G**, may be thus defined as consisting of several components:

$$\mathbf{G} \sim <\mathbf{K}, \mathbf{S}, \mathbf{T_S}, \mathbf{R}, \mathbf{N_R}, \mathbf{L}, \mathbf{M_L}> \quad (4)$$

Let the state of geographic automaton **G** at time t be S_t, located at L_t, and the external input, I_t, be defined by its neighbors R_t. The state transition, movement, and neighborhood rules—$\mathbf{T_S}$, $\mathbf{M_L}$, and $\mathbf{N_R}$—define state, location and input information of a given automaton G at time t + 1 as:

$$\begin{aligned} \mathbf{T_S}&: (S_t, L_t, R_t) \rightarrow S_{t+1} \\ \mathbf{N_R}&: (S_t, L_t, R_t) \rightarrow R_{t+1} \qquad (5) \\ \mathbf{M_L}&: (S_t, L_t, R_t) \rightarrow L_{t+1} \end{aligned}$$

Exploration with GAS then becomes an issue of qualitative and quantitative investigation of the spatial and temporal behavior of **G**, given all of the components defined above. In this way, GAS models offer a framework for considering the *spatially enabled* interactive behavior of elementary geographic objects in a system.

3.1. Tight-coupling between GAS and vector GIS

Many questions arise when applying the abstract framework mentioned above to modeling specific geographic systems. Geographic automata of different types can follow different topologies. How might we incorporate that into models: let householders react to their neighbors in the adjacent houses (irregular tessellation); drivers to cars ahead, behind, and on their sides (point events on the network); customers to shops, which can be far away (graphs)? How could automata represent continuous fields, if ever? How can theoretical ideas relating to spatial mobility—way-finding, spatial cognition, action-at-a-distance, etc.—be incorporated into the behavioral rules of the real-world mobile agents or immobile features and further translated into automation rules? How can the global patterns reflecting self-organization of non-elementary urban entities be recognized and validated if generated in simulations? Answering these questions requires a tight-coupling between GIS and automata-based models.

As a first step, we could rely on vector GIS. Indeed, vector GIS as a special kind of Database Management System (DBMS). Indeed, vector GIS may provide much support for GAS models. First and foremost, vector GIS can be used to store and retrieve the location and states of spatial objects and to register spatial actions. The next step toward fusion of GIS and geosimulation exploits an *object-based* view of urban reality. Indeed, both GAS and vector GIS are object-based in their design; both deal with discrete spatial objects, which customarily represent the real world at "microscopic" scales. A geosimulation approach considers the city as a *dynamic collective* of spatially located objects. Vector GIS deals mostly with static objects and employs an *entity-relationship model* (ERM) for their representation. In our approaches detailed here, we merge GAS and vector GIS functionality by implementing GAS as a specialized object-oriented database that manages geographic automata. Automata of a certain type are interpreted as a *class*, states and location as class *properties*; and GAS state transition, movement and neighborhood transition rules of automata of different types are thus reformulated as *methods* of corresponding automata classes. Let us specify components of GAS definition (4) – (5).

Geographic automata types, K

At an abstract level, we distinguish between *fixed* and *non-fixed* geographic automata. Fixed geographic automata represent objects that do not change their location over time and, thus, have close analogies with CA cells. In the context of urban systems, these are objects such as road links, building footprints, parks, etc. Fixed geographic automata may be subject to state and neighborhood transition rules $\mathbf{T_S}$ and $\mathbf{N_R}$, but not of rules of motion, $\mathbf{M_L}$.

Non-fixed geographic automata symbolize entities that change their location over time, for example: pedestrians, vehicles, and households. The full range of rules for GAS can be applied to non-fixed geographic automata.

Geographic automata of either fixed or non-fixed type evidently correspond to vector GIS features; normally the automata of a given type correspond to the features of a certain layer of vector GIS.

Geographic automata states and state transition rules, S and $\mathbf{T_S}$

State transition rules $\mathbf{T_S}$ are based on geographic automata of *all* types from **K**. In contrast to CA, the states **S** of urban fixed infrastructure objects depend on the neighboring objects of the infrastructure, but are also driven by non-fixed geographic automata—agents—that may be responsible for governing object states such as land-use, land value, etc. In this way, urban objects do not simply mutate like bacteria (O'Sullivan and Torrens 2000); rather, state transition is governed by other objects, the latter crucial for simulating *human-driven* urban systems, in which people are affected by their environments and also change them.

Interpreted in GIS terms, automaton states are attribute values of a corresponding GIS feature, while state transition rules exploit attributes of GIS features that are in spatial and non-spatial relationships with a given automaton.

Geo-referencing conventions, L and movement rules $\mathbf{M_L}$

Geo-referencing conventions are crucial for coupling GAS and GIS. On the one hand, they should be sufficiently flexible to enable translation of geographic perspectives on locating real world objects, both fixed and non-fixed; on the other, they should satisfy limitations of entity-relationship models, in order to be convenient for GIS management.

The GAS framework resolves these limitations by introducing two forms of geo-referencing—*direct* and *by pointing*. Direct methods of geo-referencing follow a vector GIS approach for representing reality, using coordinate lists. Such a list indicates all spatial details necessary to represent an object—automata boundaries, centroids, nodes' location, etc. Fixed geographic automata are usually located by means of direct geo-referencing. The details of the particular rules employed depend on the automata used in a modeling exercise. For typical urban objects such as buildings or street segments, 2D footprint polygons or 3D prisms may be used.

Non-fixed geographic automata may move; updating position coordinates might cause difficulties, which, when the automata are numerous and their shape is complex, are irresolvable in the framework of an entity-relationship model. We address these difficulties with a second method of geo-referencing—by *pointing* to other automata (Figure 2).

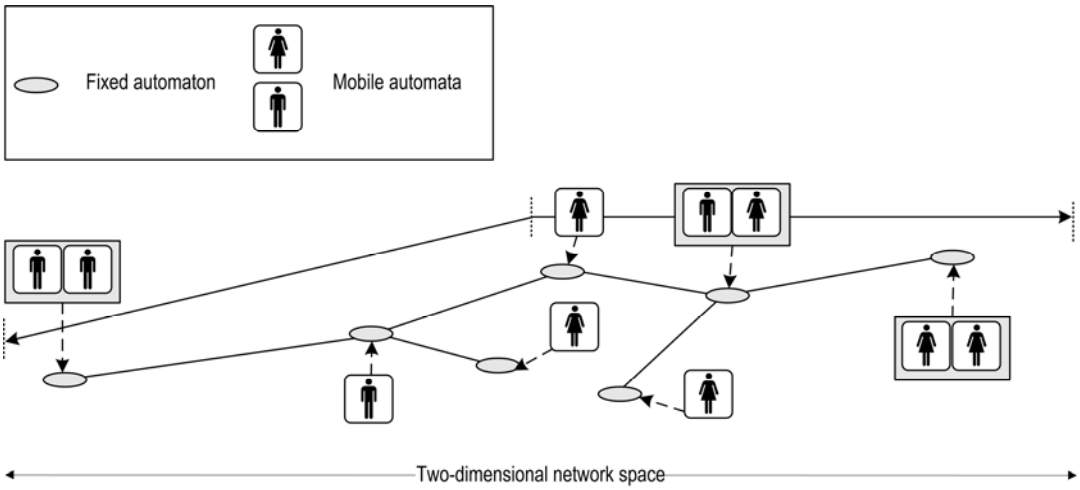

Figure 2. Direct and indirect geo-referencing of fixed and non-fixed GA

Let us consider a typical example. In the case of property dynamics, householders can be geo-referenced by address. Landlords provide a more complex case: they can be geo-referenced by pointing to the properties that they own. Indirect referencing can also be used for fixed geographic automata, e.g., for apartments in a house (Torrens 2001). Referencing by pointing is dynamic-enabled both in space and in time and compatible with the ERM database model.

GAS-based research into different formulations of $\mathbf{M_L}$ offers great potential for geographers. Realistic rules, $\mathbf{M_L}$, for encoding automata movement, based on repel-attract-synchronize interactions between close neighbors, are being developed, for example, in Animat research (Meyer et al. 2000) and in the gaming industry (Reynolds 1999). There is much opportunity for geographers to contribute to this line of research. Traditionally, attention has focused on the generation of realistic choreographies for automata, particularly in traffic models, through the specification of rules for collision avoidance, obstacle negotiation, lane-changing, flocking, behavior at junctions, etc. (Torrens 2004). However, there remain many relatively neglected areas of inquiry: spatial cognition, migration, way finding, navigation, etc.

Neighbors and neighborhood rules, R and $\mathbf{N_R}$

The set of neighbors of automata, **R**, is necessary for determining input information **I**. In contrast to the static and symmetrical neighborhoods usually employed in CA models (Figure 1a), spatial relationships between geographic automata can vary in *space and time*, and, thus, rules for determining neighborhood relationships $\mathbf{N_R}$ is necessary. Neighborhood rules for fixed geographic objects are relatively easy to define (simply because the objects are static in space). There are a variety of *geographical* ways in which neighborhood rules can be expressed for them—via adjacency of units in regular or irregular tessellations, connectivity of network nodes, proximity, etc., (Figure 1b, 1c), all fit naturally in the context of GIS operations. Spatial notions related to the incorporation of human-like automata into GAS, such as accessibility, visibility, and mental maps can be formally encoded as $\mathbf{N_R}$ rules.

Non-fixed geographic automata pose more of a challenge when specifying neighborhood rules, because the objects—and hence their neighborhood relations—are dynamic in space and time. It can be done straightforwardly, via distance and nearest-neighbor

relations, as used in Boids models (Reynolds 1987), but as movement rules $\mathbf{M_L}$. This can become very heavy computationally when complex definition of, say, visibility or accessibility is involved. In this case it is more appropriate to base neighborhood rules on indirect location, as defined in previous sections, and consider two indirectly located automata as neighbors when *the automata they point to are neighbors*.

For example, two householder agents can be established as neighbors by assessing the neighborhood relationship between the houses in which they reside. Even when these agents are physically separated in the simulation—when they go shopping or go to work, for example—they remain 'neighbors' by virtue of the (fixed) relationship between their properties.

3.2. Temporal dimension of GAS

A methodological base composed of fixed and non-fixed objects and direct and by-pointing locating implies specification of Geographic Automata Systems within GIS. However, the dynamic nature of GAS also implicates *temporal* dimensions of GIS databases, and, thus entails its own limitations. Given these considerations, transition rules $\mathbf{T_S}$, $\mathbf{M_L}$, and $\mathbf{N_R}$, should be defined in a way that avoids conflicts when geographic automata are created or destroyed and their states, locations, or neighborhood relations are updated.

According to (5), a triplet of transition rules determines the states **S**, locations **L** and neighbors **R** of automata at time t + 1 based on their values at time t. It is well known that different interpretations of the 'hidden'—time—variable in a discrete system can critically influence resulting dynamics of the model (Liu and Andersson 2004). There are several ways to implement time in a dynamic system. On the one hand, we consider time as governed by an *external* clock, which commands simultaneous application of rules (5) to each automaton and at each tick of the clock. On the other hand, each automaton can have its own *internal* clock and, thus, the units of time in (5) can have different meaning for different automata. Formally, these approaches are expressed as *Synchronous* or *Asynchronous* modes of updating of automata states and location. System dynamics depend strongly on the details of the mode employed (Liu and Andersson 2004; Berec 2002), and the spatial aspects of the problem are studied in *temporal* GIS (Peuquet 2002; Miller 1999). Tight-coupling between GAS and GIS might thus demand further development of temporal GIS.

3.3. GAS and raster GIS

Conceptually, GAS do not fit to modeling dynamics of continuous fields, as, say, erosion processes, or pollution transport. Nonetheless, it is formally easy to interpret raster GIS cells as geographic automata, if the resolution is established *a priori*. It is also worth noting that in the majority of situations the sources and recipients of continuously distributed characteristics *are* geographic automata. One can think about further coupling between GAS and continuous models, but such an extension is merely beyond the minimal framework we discuss.

4. The software implementation of GAS

A software implementation of GAS for urban modeling—Object-Based Environment for Urban Simulation (OBEUS)—is currently in development at the Environment Simulation Laboratory of the University of Tel Aviv (Benenson, Aronovich, and Noam 2001, 2004). Recently, OBEUS was modified to include all the basic characteristics of

GAS, and that was intentionally done in a minimal possible manner. Based on recent OBEUS experience, we specify—below—the main components that are necessary to implement the GAS approach as a software environment. The shareware version of OBEUS is still in development and is planned for distribution in April 2004 (see http://eslab.tau.ac.il for updates). The GAS framework is not software- or language-specific, however, and applications have been developed in different environments (Torrens 2003).

4.1. Abstract classes of OBEUS

The basic components of GAS are defined in OBEUS with respect to automata types k $\in$ **K**, its states $\mathbf{S}_k$, location **L**, and neighborhood relations **R** to other objects. These are implemented by means of the two-level structure of abstract root classes presented, partially, in Figure 3:

At the top level we define three abstract classes: **Population** contains information regarding the population of objects of given type k as a whole; **GeoAutomata** acts as a container for geographic automata of specific type; **GeoRelationship** facilitates specification of (spatial, but not necessarily) relationships between geographic automata. This functionality is available regardless of the degree of neighborhood and other relationships between automata, whether they are one-to-one, one-to-many, or many-to-many.

The *location* information for geographic automata essentially depends on whether the object we consider is fixed or non-fixed. This dichotomy is handled using abstract classes, **Estate** and **Agent**, which inherit **GeoAutomata**. The **Estate** class is used to represent fixed geographic automata . The **Agent** class represents non-fixed geographic automata.

Following from GeoRelationship, three abstract relationship classes can be specified: **EstateEstate**, **AgentEstate**, and **AgentAgent**. The latter is not implemented in order to avoid conflicts in relationship updating (see section 4.3 below), and the only way of locating non-fixed agents modeled in OBEUS is by pointing to fixed estates; consequently, direct relationships between non-fixed objects are not allowed. This might limit possible applications of the system—we already mentioned the boids model (Reynolds 1999) as an example where direct Agent-Agent relationships are necessary. At the same time, the majority of urban models we are aware of do not need them, and most examples in which Agent-Agent relationships are important arise in contexts beyond the intuitive limitations of the “reactive agent” idea; marriage is one example that a reviewer of this chapter suggested.

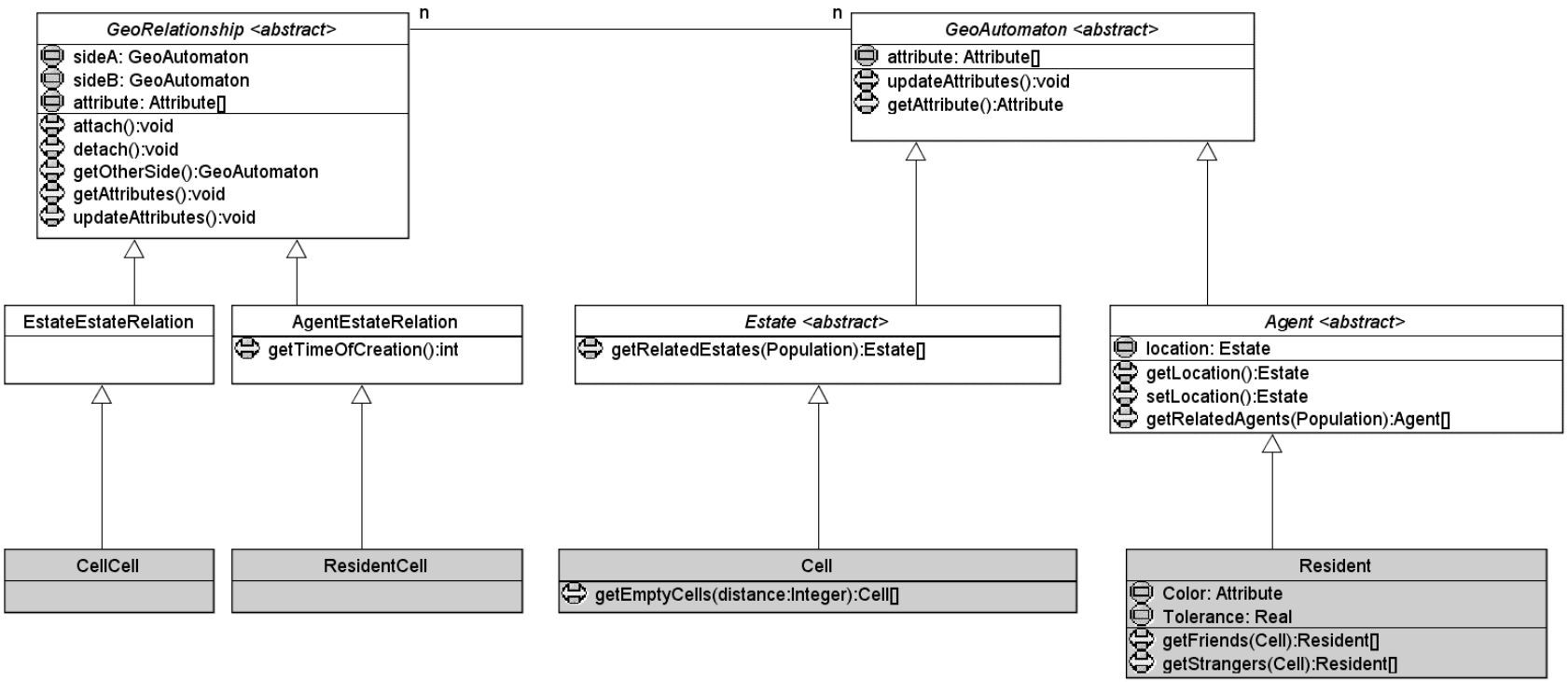

Figure 3. A UML scheme illustrating the abstract-level classes of OBEUS and the example of model-level classes for a residential dynamics simulation.

4.2. Management of time in OBEUS

The OBEUS architecture utilizes both *Synchronous* and *Asynchronous* modes of update. In *Synchronous* mode, all automata are assumed to change simultaneously and conflicts can arise when agents compete over limited resources, as in the case of two householders trying to occupy the same apartment. Resolution of these conflicts depends on the model's context, a decision OBEUS leaves to the modeler. In *Asynchronous* mode, automata change in sequence, with each observing a geographic reality left by the previous automata. Conflicts between automata are thereby resolved; but the order of updating is critical as it may influence results.

OBEUS demands that the modeler sets up an order of automata update according to a template: randomly, sequence in order of some characteristic, and object-driven approaches are currently being implemented.

4.3. Management of relationships in OBEUS

Relationships in GAS models can change in time and this might cause conflicts, when, in housing applications, for example, a landlord wants to sell his property, while the tenant does not want to leave the apartment. This example represents the general *problem of consistency* in managing relationships. It has no single general solution; there are plenty of complex examples discussed in the computer science literature (Peckham, MacKellar, and Doherty 1995). OBEUS aims at minimal representation of the GAS framework and thus follows the leader-follower development pattern proposed by Noble (2000). To maintain consistency in relationships, an object on one side, termed the *leader*, is responsible for managing the relationship. The other side, the *follower*, is comprised of passive objects. The leader provides an interface for managing the relationship, and invokes the followers when necessary. There is no need to establish leader or follower 'roles' in a relationship between fixed objects once the relationship is established, while in relationships between a non-fixed and a fixed object, the non-fixed object is always the leader and is responsible for creating and updating the relationship. For instance, in a relationship between a landlord and her property (when ownership cannot be shared), the landlord initiates the relationship and is able to change it. We do not have evidence that the majority of real-world situations can be imitated using a *leader-follower* pattern, but we are also unaware of cases—in urban contexts—where this pattern would be insufficient.

4.4. Implementing system theory demands within OBEUS

Systems theory suggests another challenge for automata modeling in which the usefulness of the GAS-OBEUS approach appears to offer advantages. In a systems context, *many* interacting automata are often necessary for capturing the nuances of geographic reality. It is very well known that if system rules are non-linear and the system is open, then emergence and self-maintenance of entities at above-automata levels become feasible. Gentrified areas and commuting zones are examples.

The idea of self-organization is external to GAS and it is not necessary to incorporate it into the software implementation. Nonetheless, self-organization is often too important for studying urban systems to be ignored, even at the first step of GAS software implementation. To accommodate this, emerging *spatial ensembles* of geographic automata are supported in OBEUS by means of the abstract class **GeoDomain**. The simplest approach to emergence, determined by the set of *a priori* given predicates

defined on geographic automata is implemented; domains are thus limited to capturing 'foreseeable' self-organization of specific types.

As with non-fixed agents, domains are always leaders in their relationships with the automata within them. These relationships can capture properties such as distance between fixed automata and the domain; several definitions of distance based on objects' and domains' centroids, boundaries, etc. can be applied.

4.5. Example of GAS modeling style – the Schelling model in terms of OBEUS

The well-known Schelling model of residential dynamics (Hegelsmann and Flache 1998; Schelling 1971) reflects a basic idea of residential segregation between the members of two mutually avoiding population groups, as a self-organizing phenomenon. Thomas Schelling realized his model with black and white checkers on a chess-board and from then on, the model is formulated in terms of agents of two types, B and W, located in the cells of a regular grid, with a maximum of one agent per cell. The essence of the Schelling model is in assumption that if the fraction of agents of strange type (say, of W-type for the B-agent) within the neighborhood of a current agent's location is above the tolerance threshold of an agent, then an agent tries to relocate to a nearest unoccupied location, where the fraction of strange agents is below that threshold. The Schelling model is asynchronous and each agent observes the state of the system as left by the previous one.

As an applied example of GAS, let us formulate this model in GAS terms and then present the entities and methods necessary to implement it in OBEUS.

Two **types** of objects (K = 2)
Fixed Objects: Cells C;
Cell states S:
Cells have no states, i.e. $S_1 = Ø$;
Cell location rules L:
Each cell C is assigned a pair of coordinates (x, y) for direct location,
L_1 = {Locate C(x, y) at (x, y)};
Neighborhood rules R (Moore, as an example):
R_1 = {For a cell C(x, y), cells B(u, v), given max(|u – x|, |v – y|) = 1 are neighbors};
State transition rules T:
None, i.e. $T_1 = Ø$;
Movement rules M:
None, $M_l = Ø$, there are no movement rules for fixed objects;
Neighborhood transition rules N:
None, $N_1 = Ø$, neighborhood relations of fixed objects do not change.

Non-Fixed Objects: Residents D,
Resident states S:
Resident can be in one of two states, B or W, S_2 = {B, W}
Resident location rules L:
Resident is located indirectly, by pointing to a cell it occupies, R_2 = {Cell C}
Neighborhood rules R:
R_2 = {Residents located in neighboring cells are neighbors}

State transition rules T:

None, $T_2 = Ø$;

Neighborhood transition rules N:

None, $N_2 = Ø$.

Movement rules (one of the versions) M:

M_2 = {If fraction of strangers f_R among the neighbors is below threshold, i.e. $f_R < f_{TH}$, do nothing; otherwise find the closest unoccupied cell satisfying $f_R < f_{TH}$ and relocate there; if there is more than one at the same distance choose one among them randomly; if no suitable location is found do nothing}

The OBEUS implementation of the 'Schelling' GAS is as follows (Figure 3)

1. The abstract classes **Agent** and **Estate** are inherited by ***Resident*** and ***Cell***, respectively;
2. The abstract relationship classes **EstateEstate** and **AgentEstate** are inherited by ***CellCell*** and ***ResidentCell***, respectively;
3. Class ***Cell*** has no properties of its own;
4. Class ***Cell*** has one basic method of its own ***getEmptyCloseCells(distance, cell)***, which employs **getRelatedEstates (estate)** and returns the list of non-occupied neighboring cells at a distance ***distance***.
5. Class ***Resident*** has two properties of its own - Boolean ***color***, with values B and W, and real ***Tolerance*** threshold;
6. Class ***Resident*** has two basic methods of its own, ***getFriends(resident, cell)*** and ***getStrangers(resident, cell)***, which employ **getRelatedEstates(estate)**, and return lists of the neighbors of the color which is same or opposite to the color of a ***resident*** if located in a cell ***cell***.

The goal of the Schelling GAS is to study residential segregation and, thus, its investigation demands recognition of areas populated mostly or exclusively by B- or W-agents. This can be done by specifying domain criteria based on the ***CellCell*** and ***ResidentCell*** relationship. Namely, let us consider a cell C occupied by resident R of a color L. Let us define the cell C as S_L-true (segregation-true of a color L) if the fraction of resident's neighbors of color L is above the given threshold f. For the Schelling model, if we take the value of f high enough (and higher than the tolerance threshold of residents), the sets of S_L-true cells for L = B and L = W will form continuous areas that reflect intuitive understanding of segregation. The value of f determines the density of the residents of the same color in the domain and the degree of the overlap of S_B and S_W domains. The concept of domains is considered in more details in (Benenson, Aronovich, and Noam 2004) and it is beyond the scope of this paper. Let us note, however, that we consider the formulation of domain criteria and investigation the patterns produced by the geographic automata that satisfy the criteria as a necessary step in understanding spatial patterns that can emerge in the model.

5. Conclusions

We have introduced a GAS framework as a unified scheme for representing discrete geographic systems. Technically, the framework is designed to merge two popular tools used in urban simulation—CA and MAS —and specify them in a patently spatial manner. Conceptually, our assertion is that GAS forms the kernel of the system, as far as the system is spatially driven.

The minimal GAS skeleton allows for a degree of standardization between automata models and GIS. It also provides a mechanism for *transferability*. Until now, the majority of spatial simulations can be investigated only by their developers. The development of GAS software breaches this barrier, offering opportunities to turn urban modeling from art into engineering.

A few additional steps are necessary for full implementation of the GAS framework; none, we think, requires decades of research to realize. The first requirement we have identified relates to transforming the GAS framework into software. As demonstrated with reference to OBEUS, we have advanced along that line of research inquiry, in urban contexts. Development of a (preferably geography-specific) simulation language based on GAS is a second requirement that we consider. The intent, in that context, is to enable the formulation of simulation rules in terms of objects' spatial behavior. We believe that the continued development of simulation languages (Schumacher 2001) that has gathered steam in the last decade, coupled with advances in GI Science and spatial ontology, could answer this requirement in the near future. The third requirement is development of GAS applications. We have developed GAS-based models of housing dynamics (Benenson, Omer, and Hatna 2002) and urban growth (Torrens 2003, 2002) thus far, and the results are promising.

6. References

Albin, Peter. S. 1975. *The Analysis of Complex Socioeconomic Systems*. Lexington. MA: Lexington Books.

Batty, Michael, Helen Couclelis, and M. Eichen. 1997. Special issue: urban systems as cellular automata. *Environment and Planning B* 24 (2).

Benenson, Ithzak, and Paul M. Torrens. 2004. *Geosimulation: Automata-Based Modeling of Urban Phenomena*. London: John Wiley & Sons.

Benenson, Itzhak. 1998. Multi-agent simulations of residential dynamics in the city. *Computers, Environment and Urban Systems* 22 (1):25-42.

———. 1999. Modelling population dynamics in the city: from a regional to a multi-agent approach. *Discrete Dynamics in Nature and Society* 3:149-170.

Benenson, Itzhak, S. Aronovich, and S. Noam. 2001. OBEUS: Object-Based Environment for Urban Simulations. In *Proceedings of the Sixth International Conference on GeoComputation*, edited by D. V. Pullar. Brisbane: University of Queensland, GeoComputation CD-ROM.

———. 2004. Let's talk objects. *Computers, Environment and Urban Systems* (forthcoming).

Benenson, Itzhak, and Itzhak Omer. 2000. Multi-scale approach to measuring residential segregation and the case of Yaffo, Tel-Aviv. In *Proceedings of the Fifth Annual Conference on GeoComputation*, edited by R. Abrahart. Manchester: GeoComputation CD-ROM.

———. 2003. High-resolution Census data: a simple way to make them useful. *Data Science Journal (Spatial Data Usability Special Section)* 2 (26):117-127.

Benenson, Itzhak, Itzhak Omer, and Erez Hatna. 2002. Entity-based modeling of urban residential dynamics: the case of Yaffo, Tel Aviv. *Environment and Planning B: Planning and Design* 29:491- 512.

Benenson, Itzhak, and Paul M. Torrens. 2004. Geosimulation: object-based modeling of urban phenomena. *Computers, Environment and Urban Systems* 28 (1/2):1-8.

Berec, L. 2002. Techniques of spatially explicit individual-based models: construction, simulation, and mean-field analysis. *Ecological Modelling* 150:55-81.

Box, Paul. 2001. Spatial Units as Agents: Making the Landscape an Equal Player in Agent-Based Simulations. In *Integrating Geographic Information Systems and Agent-Based*

Modeling Techniques for Simulating Social and Ecological Processes, edited by H. R. Gimblett. Oxford: Oxford University Press.
Brown, John Seely, and Paul Duguid. 2000. *The Social Life of Information*. Boston: Harvard Business School Press.
Chapin, F. Stuart, and Shirley F. Weiss. 1968. A Probabilistic Model for Residential Growth. *Transportation Research* 2:375-390.
Couclelis, Helen. 1985. Cellular worlds: A framework for modeling micro-macro dynamics. *Environment and Planning A* 17:585-596.
Dijkstra, J., H.J.P. Timmermans, and A.J. Jessurun. 2000. A multi-agent cellular automata system for visualising simulated pedestrian activity. In *Theoretical and Practical Issues on Cellular Automata*, edited by S. Bandini and T. Worsch. London: Springer-Verlag.
Epstein, Joshua M. 1999. Agent-based computational models and generative social science. *Complexity* 4 (5):41-60.
Epstein, Joshua M., and Robert Axtell. 1996. *Growing Artificial Societies from the Bottom Up.* Washington D.C.: Brookings Institution.
Gimblett, H. Randy, ed. 2002. *Integrating Geographic Information Systems and Agent-Based Modeling Techniques for Simulating Social and Ecological Processes, Santa Fe Institute Studies in the Sciences of Complexity.* Oxford: Oxford University Press.
Hegelsmann, Rainer, and Andreas Flache. 1998. Understanding complex social dynamics: a plea for cellular automata based modelling. *Journal of Artificial Societies and Social Simulation* 1 (3).
Kohler, Timothy A. 2000. Putting social sciences together again: an introduction to the volume. In *Dynamics in Human and Primate Societies*, edited by T. A. Kohler and G. Gumerman. New York: Oxford University Press.
Liu, Xiao-Hang, and Claes Andersson. 2004. Assessing the impact of temporal dynamics on land-use change modeling. *Computers, Environment and Urban Systems* 28 (1/2):107-124.
Meyer, Jean-Arcady, Alain Berthoz, Dario Floreano, Herbert L Roitblat, and Stewart W Wilson, eds. 2000. *From Animals to Animats 6: Proceedings of the Sixth International Conference on Simulation of Adaptive Behavior*. Cambridge, MA: MIT Press.
Miller, Harvey J. 1999. Measuring space-time accessibility benefits within transportation networks: basic theory and computation procedures. *Geographical Analysis* 31 (2):187-213.
Nakajima, Takashi. 1977. Application de la théorie de l'automate à la simulation de l'évolution de l'espace urbain. In *Congrès Sur La Méthodologie De L'Aménagement Et Du Dévelopment*. Montreal: Association Canadienne-Française Pour L'Avancement Des Sciences et Comité De Coordination Des Centres De Recherches En Aménagement, Développement Et Planification (CRADEP).
Noble, J. 2000. Basic relationship patterns. In *Pattern Languages of Program Design 4*, edited by N. Harrison, B. Foote and H. Rohnert. New York: Addison-Wesley.
O'Sullivan, David. 2001. Exploring spatial process dynamics using irregular cellular automaton models. *Geographical Analysis* 33 (1):1-18.
O'Sullivan, David, and Paul M. Torrens. 2000. Cellular models of urban systems. In *Theoretical and Practical Issues on Cellular Automata*, edited by S. Bandini and T. Worsch. London: Springer-Verlag.
Peckham, J., B. MacKellar, and M. Doherty. 1995. Data models for extensible support of explicit relationships in design databases. *VLDB Journal* 4:157-191.
Peuquet, Donna J. 2002. *Representations of Space and Time*. New York: Guilford.
Phipps, M. 1989. Dynamic behavior of cellular automata under the constraint of neighborhood coherence. *Geographical Analysis.* 21:197-215.
Polhill, J. G., N. M. Gotts, and A. N. R. Law. 2001. Imitative and non-imitative strategies in a land use simulation. *Cybernetics and Systems* 32:285-307.
Portugali, J. 2000. *Self-Organization and the City*. Berlin: Springer-Verlag.
Portugali, J., I. Benenson, and I. Omer. 1997. Spatial cognitive dissonance and sociospatial emergence in a self-organizing city. *Environment and Planning B* 24:263-285.

Portugali, Juval, Itzhak Benenson, and Itzhak Omer. 1994. Socio-spatial residential dynamics: stability and instability within a self-organized city. *Geographical Analysis* 26 (4):321-340.
Reynolds, Craig. 1987. Flocks, herds, and schools: A distributed behavioral model. *Computer Graphics* 21 (4):25-34.
———. 1999. Steering behaviors for autonomous characters. Paper read at Game Developers Conference, at San Jose, CA.
Sakoda, J.M. 1971. The checkerboard model of social interaction. *Journal of Mathematical Sociology* 1:119-132.
Sanders, L, D Pumain, H Mathian, F Guérin-Pace, and S Bura. 1997. SIMPOP: A multiagent system for the study of urbanism. *Environment and Planning B* 24:287-305.
Schelling, Thomas C. 1969. Models of segregation. *American Economic Review* 59 (2):488-493.
———. 1971. Dynamic models of segregation. *Journal of Mathematical Sociology* 1:143-186.
———. 1974. On the ecology of micro-motives. In *The Corporate Society*, edited by R. Marris. London: Macmillan.
———. 1978. *Micromotives and Macrobehavior*. New York: WW Norton and Company.
Schofisch, B., and K.P. Hadeler. 1996. Dimer automata and cellular automata. *Physica D* 94:188-204.
Schumacher, M. 2001. *Objective Coordination in Multi-Agent System Engineering*. Berlin: Springer.
Semboloni, Fernando. 2000. The growth of an urban cluster into a dynamic self-modifying spatial pattern. *Environment and Planning B: Planning & Design* 27 (4):549-564.
Shi, Wenzhong, and Matthew Yick Cheung Pang. 2000. Development of Voronoi-based cellular automata--an integrated dynamic model for Geographical Information Systems. *International Journal of Geographical Information Science* 14 (5):455-474.
Tobler, Waldo. 1970. A computer movie simulating urban growth in the Detroit region. *Economic Geography* 46 (2):234-240.
———. 1979. Cellular Geography. In *Philosophy in Geography*, edited by S. Gale and G. Ollson. Dordrecht: Kluwer.
Torrens, Paul M. 2001. New tools for simulating housing choices. Berkeley, CA: University of California Institute of Business and Economic Research and Fisher Center for Real Estate and Urban Economics.
———. 2002. Cellular automata and multi-agent systems as planning support tools. In *Planning Support Systems in Practice*, edited by S. Geertman and J. Stillwell. London: Springer-Verlag.
———. 2002. SprawlSim: modeling sprawling urban growth using automata-based models. In *Agent-Based Models of Land-Use/Land-Cover Change*, edited by D. C. Parker, T. Berger, S. M. Manson and W. J. McConnell. Louvain-la-Neuve, Belgium: LUCC International Project Office.
———. 2003. Automata-based models of urban systems. In *Advanced Spatial Analysis*, edited by P. A. Longley and M. Batty. Redlands, CA: ESRI Press.
———. 2004. Geosimulation approaches to traffic modeling. In *Transport Geography and Spatial Systems*, edited by P. Stopher, K. Button, K. Haynes and D. Hensher. London: Pergamon.
———. 2004. Looking forward: remote sensing as dataware for human settlement simulation. In *Remote Sensing of Human Settlements*, edited by M. Ridd. New York: John Wiley and Sons.
Torrens, Paul M., and David O'Sullivan. 2000. Cities, cells, and cellular automata: Developing a research agenda for urban geocomputation. In *Proceedings of the Fifth Annual Conference on GeoComputation*, edited by R. Abrahart. Manchester: GeoComputation CD-ROM.
———. 2001. Cellular automata and urban simulation: where do we go from here? *Environment and Planning B* 28 (2):163-168.
Wahle, J., L. Neubert, J. Esser, and M. Schreckenberg. 2001. A cellular automaton traffic flow model for online simulation of traffic. *Parallel Computing* 27:719-735.

White, Roger, Guy Engelen, and Inge Uljee. 1997. The use of constrained cellular automata for high-resolution modelling of urban land use dynamics. *Environment and Planning B* 24:323-343.

Woolridge, M.J., and N.R. Jennings. 1995. Intelligent agents: theory and practice. *Knowledge Engineering Review* 10 (2):115-152.