Dynamically Tracking the Real World in a CSS Model

H. Van Dyke Parunak[1], S. Hugh Brooks[2], Sven Brueckner[1], Ravi Gupta[3]

[1]Jacobs Technology, 3520 Green Court, Suite 250, Ann Arbor, MI 48105
{van.parunak, sven.brueckner}@jacobs.com
[2]P.O. Box 25275, Washington, DC 20027
shb32@georgetown.edu
[3]enkidu7, 115 Oronoco Street, Alexandria, Virginia 22314
ravi@enkidu7.com

Abstract. CSS models are most commonly used to articulate theories and explore their implications. As they become more mature, they are also valuable in monitoring real-world situations. Such applications require models to be linked to dynamic real-world data in real time. This paper explores this distinction in a specific application that tracks crowd violence in an urban setting.

Keywords: forecasting, calibration, apoptosis, stigmergy

1 Introduction

On May 20, 2012, NATO held a summit in Chicago, IL. Protesters planned a demonstration. They registered with the authorities for a permit, which specified a route ending near the secure area where NATO delegates and heads of state would be meeting. The Cook's County Sheriff's office invited NEK Advanced Security Group to demonstrate the usefulness of social media in tracking unrest, and NEK invited enkidu7[1] to demonstrate how agent-based modeling could help monitor the demonstration in real-time and give near-term forecasts of possible "hot spots" requiring additional police attention. In response, we constructed and demonstrated a prototype of CAVE (Crowd Analysis for Violence Estimation).

Crowd simulation is an important and fairly mature area of computational social science (CSS). We do not offer any theoretical advances over previous research, from which we borrow liberally. However, we do apply these techniques in a novel way. In a research setting, CSS models serve to articulate a theory in a precise way, and (calibrated with static input data) to test the theory against historical observations. CAVE must continuously update itself with real-world data to provide an ongoing estimate of the state of the world a short distance into the future. Our contribution is demonstrating practical techniques for tracking the real world with a computational model.

Section 2 of this paper distinguishes three applications of CSS: theory articulation, static prediction, and real-time monitoring and forecasting. Section 3 briefly reviews

[1] www.enkidu7.com

the particular CSS model that we adapt, summarizes its structure and operation (specified more fully in the ODD appendix), and explains how we interface it with the real world. Section 4 reports the behavior of CAVE during the NATO summit. Section 5 concludes the main body of the paper. Section 6 presents the ODD protocol for CAVE, which provides further details on its structure and operation.

2 CSS for Theory, Prediction, and Monitoring

CSS models can be applied in several different ways. We distinguish three.

2.1 Theory Articulation

A CSS model provides an unambiguous expression of the interaction of various causal influences within and among actors in a social scenario. It is a detailed embodiment of claims about factors and interactions that in a previous era could only be outlined verbally. A number of different formalisms for such models have been demonstrated [3]. We focus on agent-based models, which represent individuals (or small groups of individuals) as software agents [14]. Valuable insights concerning social phenomena can be gleaned from interview protocols (e.g., [15,16]), but the resulting theories are difficult to test. A computational model is not only more precise than a verbal theory, but it also allows testing of hypotheses by executing the model. Even without external data, it can demonstrate testable qualitative trends and emergent behaviors that are not obvious from a verbal statement. For example, in the study on which CAVE is based [8], the tendency of groups to form as a function of crowd size is markedly different with two populations of different sizes than with balanced populations, and the emergence of violence depends on the size of the overall crowd.

2.2 Static Prediction

Qualitative agreement between simulation and observation is good, but accurate quantitative predictions (e.g., [1]) are even better, since their results are more directly comparable with observations from the real world, and they can support decisions that depend on a quantitative trade-off between cost and benefit. The first benefit is seen in an implementation that ingests live data at one point in time, then compares model outputs with subsequent observations. The second is clear in "what-if" exercises, in which the user runs the model off-line, then examines its results to guide a decision.

2.3 Real-Time Monitoring

Like static prediction models, CAVE seeks to align itself with data from the real world. However, it runs on-line, not off-line. It continuously ingests observed data and adjusts its configuration to give the user a continuously updated short-horizon forecast of the system being modeled.

Our approach is motivated by a fundamental limitation of predicting complex non-linear systems. The farther one seeks to project the dynamics of the system, the more random the projection becomes, resulting in a "prediction horizon" beyond which such a prediction is no better than random. We have demonstrated this horizon in simple agent-based models [12]. The limitation is fundamental in nature, not due to noise in the input data or shortcomings in model accuracy [17].

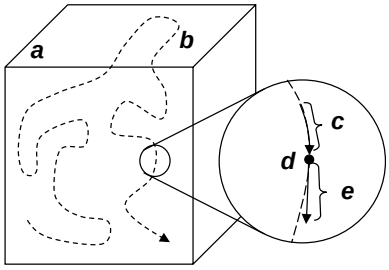

Fig. 1. Real-Time Monitoring of Complex Trajectories.—*a*: state space. *b*: system trajectory. *c*: recent observed system state. *d*: model update. *e*: short-range forecast

Abstractly, we can view the system as a vector differential equation,

$$\frac{d\vec{x}}{dt} = f(\vec{x})$$

When f is nonlinear, long-range prediction is impossible. However, it is often useful to anticipate the system's behavior a short distance into the future. A common technique is to fit a convenient low-order form for f to the system's trajectory in the recent past, and then extrapolate this fit into the future (Fig. 1, [9]). Iterating this process provides the user with a limited look-ahead into the system's future. The process is like walking through the woods on a moonless night. The traveler cannot see the other side of the forest, but her flashlight can show her the next few meters, and when she has covered that distance, it can show her the next few meters beyond that.

Realizing the program of Fig. 1 directly requires specifying the state space of the system explicitly, writing a set of differential equations that characterize it, and fitting an analytical function to recent observations. Agent-based modeling is attractive for social systems just because it is difficult to define the complete state space and express the system's behavior in terms of analytical functions. Thus it is difficult to use this technique to produce a fit. This paper shows how to approximate the strategy of Fig. 1 in an agent-based social simulation.

To motivate our approaches, let's look in more detail at local approximations to the system's state trajectory (Fig. 2). At time t_1, we fit a linear model *a*. At a subsequent time $t_2 > t_1$, we fit model *b*. These two models differ in two ways, each of which leads to errors. We can use observational data to correct both kinds of error.

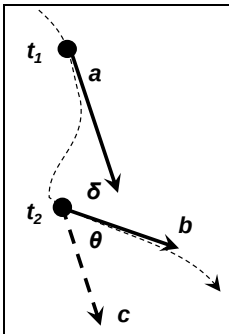

Fig. 2. Two adjustments in real-time monitoring.—Correcting system location (δ) and model fit (θ) at two time instances $t_1 < t_2$.

1. They differ in *direction*, which in this case corresponds to the internal structure and parameters of the model. The direction of the later fit *b* differs from that of *a* by θ.
2. They differ in *origin*. Model *a*, an approximation, experiences an error δ with respect to the real system.

The simplest use of observational data is simply to restart the (original) model at the new, observed location, yielding

model *c*. If the model parameters are not completely off the mark, the model still moves in the same general direction as the system.

In addition to reinitializing the model, we can also retune its parameters. When analytical approaches are not applicable, we use synthetic evolution. Fig. 3 illustrates the polyagent approach [13], representing each real-world entity by a single persistent *avatar* and a swarm of *ghosts*. The avatar continuously inserts a stream of simple agents in a faster-than-real-time model of the environment, a short distance in the past, and evolves their behavioral parameters until they correspond to observed behavior, then lets them run into the future to generate a prediction. The ghosts are apoptotic: they die after a specified period, so the system does not become clogged with an increasing number of agents.

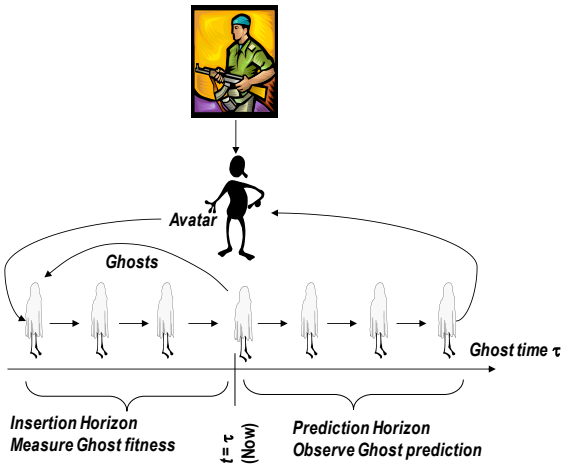

Fig. 3. Polyagent mechanism for dynamically learning agent behavioral parameters

We have demonstrated this approach in combat modeling [11]. While effective, it requires detailed observations of each entity being modeled in order to tune the ghosts' behaviors. In CAVE, we have aggregate observations of crowd size and composition, but not individual observations. So we do not evolve agents' behavioral models, and do not maintain the multiple representations of the world at different epochs required by the polyagent model. The CAVE approach resembles vector *c* in Fig. 2. Agent execution provides a short-range look-ahead into the future, while apoptosis limits the depth of the look-ahead and allows us to reinitialize agents based on real-world data, shifting the origin (though not the parameters) of the agents.

3 The CAVE Model

CAVE draws on existing models of crowd psychology, using apoptosis and real-time data acquisition to adjust the model continuously.

3.1 Underlying CSS Theory

CAVE draws on two areas of research in crowd dynamics.

First, from the extensive literature on crowd psychology [21], we use the extended social identity model (ESIM) [8,15,16]. Unlike many other models, it is extensively supported by real-world evidence. While ESIM is not restricted to aggressiveness or

violence, it has lent itself to several previous agent-based models of these behaviors [2,8,20] from which we draw inspiration. We adopt two conventions from [8].

1. Agents' aggressive behavior is driven largely by an internal state variable ([8]'s "aggression motivation") that in turn is influenced by events around them. In CAVE, this variable is called "Disrespect."
2. Agents are not homogeneous, even within one side of a two-sided conflict, but differ in their degree of commitment to the cause.

Second, there is increasing anecdotal evidence that agitators in public events use network technology such as instant messaging and other social media in real-time to coordinate their activities [19], and that the contents of such media can be analyzed to track crowd sentiment [4].

An important feature of our work (like that of reference [8], are simple rule-based entities without an elaborate model of individual cognition.

3.2 Model Structure

Fig. 4 summarizes the overall information flow in CAVE. The following sections discuss the regions of the Figure, and the ODD protocol (Section 0) gives more detail.

The Environment.
The environment (bottom of Fig. 4) is a square lattice with cells 40 m on a side, representing downtown Chicago, derived from a GIS map.[2] We label each cell to indicate whether it contains a road, the approved protest route, the security zone within which the summit activities take place, and an extra-high security exclusion zone. The shading in Fig. 5 shows the cells corresponding to each of these categories. Agents are only created on roads. They can move off of roads, but their stigmergic interactions are limited to roads, and in no case can they enter the Security or Exclusion zones.

Human Agents: Behavioral Loop.
CAVE has two types of agents representing humans: protesters and police. These agents execute two loops. This section discusses the behavioral loop (at the bottom of Fig. 4), with a frequency ω of once per simulation step. The next section discusses the lifetime loop, which implements agent apoptosis.

Protesters are of three subtypes.

- Anarchists aggressively seek to disrupt society, and energize their followers via social media. They can often be identified visually: they often wear bulky clothing to conceal hidden weapons, and also sometimes organize "black blocks," wearing black clothing and ski masks and moving cohesively to advertise their unified strength. Black blocks are known for engaging in violence and inciting clashes with the police.
- Followers accept the anarchist's agenda, but are not leaders.

[2] https://data.cityofchicago.org/browse?tags=shapefiles

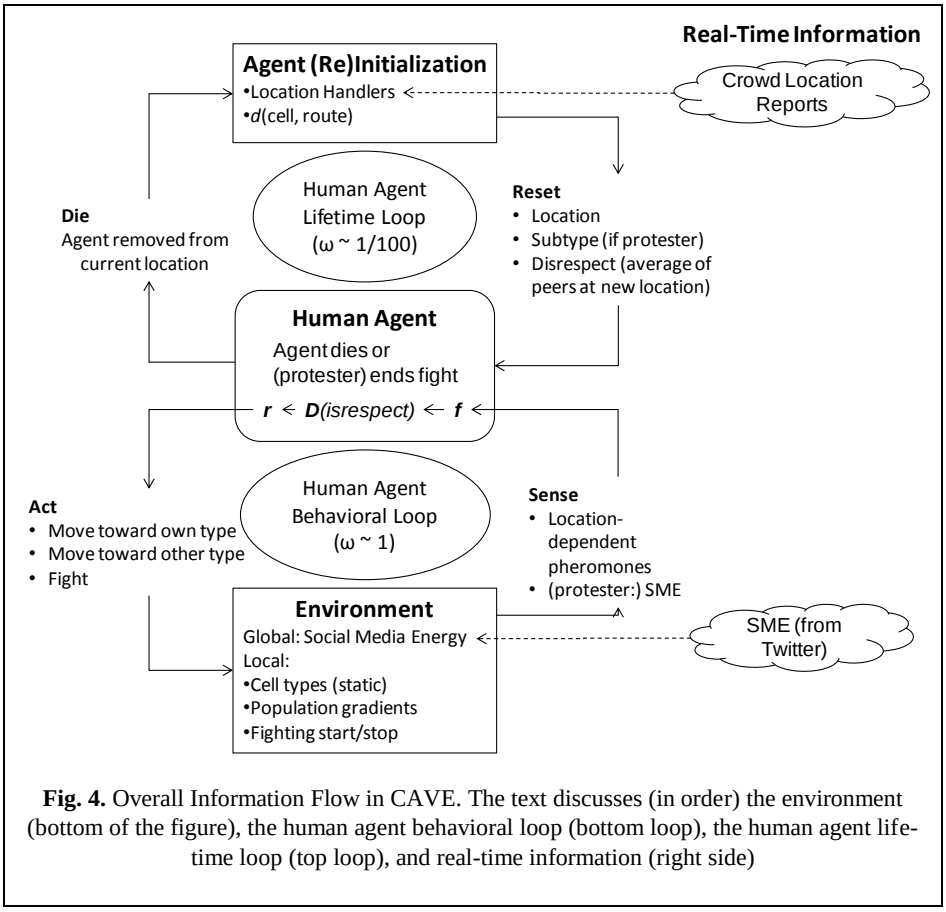

Fig. 4. Overall Information Flow in CAVE. The text discusses (in order) the environment (bottom of the figure), the human agent behavioral loop (bottom loop), the human agent life-time loop (top loop), and real-time information (right side)

- Pacifists are following the protesters out of curiosity more than ideology.

Police are of two types.

- Patrol officers are the usual cadre of an urban police force.
- Riot police have special training in dealing with unruly crowds, as well as specialized equipment such as riot shields and heavier padded armor.

The user initializes the total number of protesters and police, and the subtypes are allocated according to fixed proportions that are model parameters.

At the beginning of a run, the agents are distributed randomly on the roads in the environment that are outside the Security and Exclusion zones. Each road cell has a probability of receiving an agent that depends on how far the cell is from the protest route. The initialization function for protester agents concentrates them on the protest route, that for anarchists lets them wander farther than pacifists, and that for police keeps them near the protest route but not blocking traffic by being on it (Section 6.5).

Each agent's behavior is driven by its level of Disrespect, a variable that is defined by its drivers and its consequences. An agent's Disrespect increases with the presence

Fig. 5. Coding of the CAVE Environment

of fighting in its cell, and (in the case of protesters) the level of Social Media Energy (SME) attested by tweets from the anarchists. In the bottom loop of Fig. 4, an agent's input *f*(unction) translates the environmental state that it senses into a level of *D*(isrespect), which is then translated via a *r*(ule) into one of three actions. The consequences of increased Disrespect are that the agent first moves toward other agents of its own type (Protester or Police) for protection, then moves toward agents of the opposite type (in preparation for confrontation), and then engages in a fight.

Agents execute in random order, without replacement within a given simulation step. A single step corresponds to one minute of real-world time; the actual elapsed time is much less, and depends on the speed of the processor.

Agents interact, not directly, but through a shared environment in which they are localized. The environment is not passive, but executes some processes that support the agent coordination. This pattern of coordination is called "stigmergy," a biological term that recalls the social insects by which the mechanism is inspired [5,10].

Each time it executes, an agent deposits digital pheromones on its cell in the environment, indicating its type, its presence, its level of Disrespect, and whether it is engaged in Fight behavior. Similarly, agents sense a fight in the neighborhood by monitoring Fight pheromone, and move toward other agents of a specified type by climbing the gradient of the presence pheromone associated with agents of that type.

The environment supports pheromone-based interaction by evaporating all pheromones exponentially, thus removing obsolete information from the system. In AI terms, it provides basic truth maintenance (maintaining the consistency of a database), a task that is NP-complete in symbolic representations, with time complexity $O(1)$.

Human Agents: Lifetime Loop.

Apoptotic agents are central to CAVE's real-time updating. Each agent's lifetime is assigned when it is created from a uniform distribution on [50, 150]. When its lifetime is over, the agent is reinitialized to another location, based on real-time observations

of the distribution of protesters and police. The top loop of Fig. 4 summarizes this life-cycle, whose frequency ω is on the order of 1/100.

Apoptotic agents address two challenges facing an agent model that seeks to be aligned with the real world.

1. How does the model adjust its internal state to stay aligned with the real world (Fig. 2)?
2. How do we manage the relation between the simulation's internal clock (which depends on processor speed) and the dynamics of the real world?

The reassignment of agents to new locations at the end of their life addresses the first challenge. Each time CAVE receives an observation of a concentration of protesters or police, it instantiates a special agent (a "location handler") at the location of the observation. If the current population of agents at a location is greater than the location handler desires, it inhibits the assignment of reinitialized agents to that location, and apoptosis eventually reduces the population to the observed level. If the current population is too low, the location handler attracts reinitialized agents to its location. Thus, with a half-life of 100 minutes (the mean of the lifetime assignment distribution), population levels in the model adjust to match observed levels.

Apoptosis also mitigates the problem of varying execution speed. The mean agent lifetime is 100 minutes. Because lifetimes are randomized in [50, 100], agents are reborn at different times. After a few hundred steps, the average agent has been active for about 53 steps (Fig. 6), and the strength of the estimated violence reflects a lookahead about this distance into the future. With a modern computer, a single simulation step takes only a few milliseconds, so the view on the display is looking roughly 53 minutes into the future. Agent apoptosis keeps them from running indefinitely into the future and formulating an unjustified long-range forecast.

Data Sources.

CAVE is continuously updated with two real-world data sources, shown on the right-hand side of Fig. 4: an estimate of *SME* from Twitter feeds (modulating the behavioral loop), and estimates of crowd density from human observers (modulating the lifetime loop).

Anarchists use social media such as Twitter to communicate with their followers. The effectiveness of this communication mechanism depends on their Twitter handles and relevant Twitter hashtags being known, so police can monitor their tweets. CAVE processes this stream of tweets through a

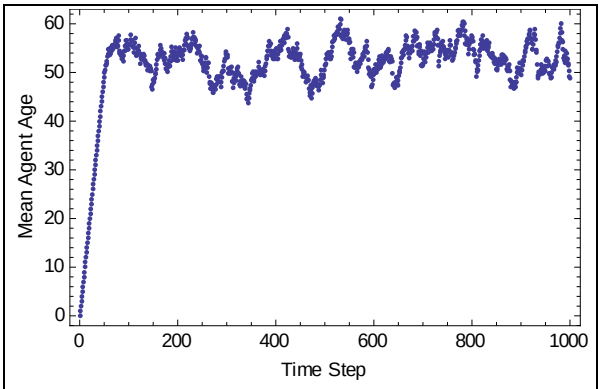

Fig. 6. Mean agent age = average lookahead

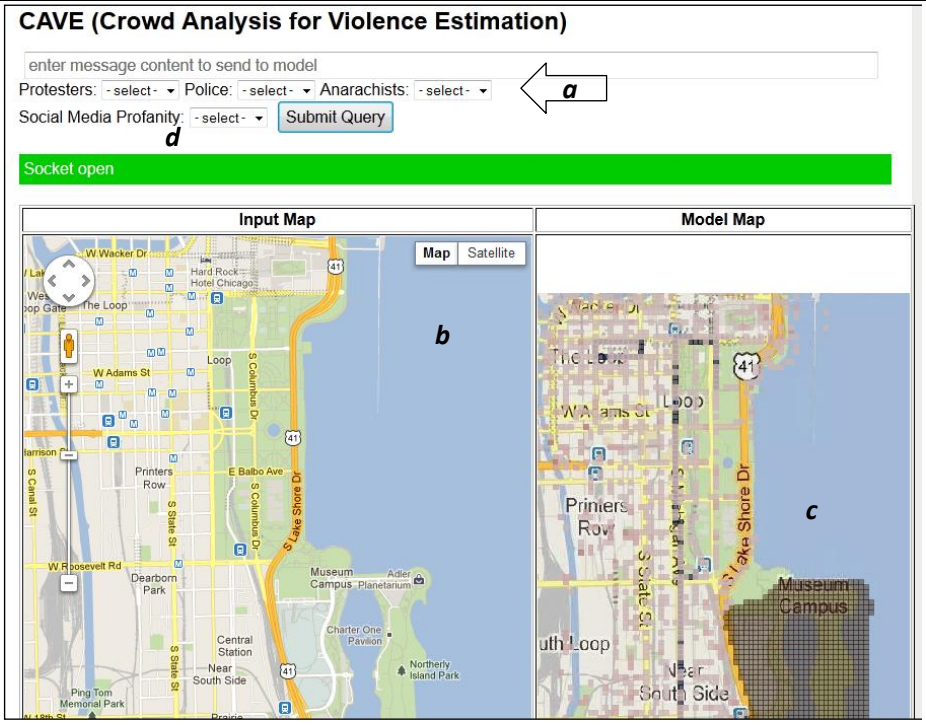

Fig. 7. CAVE interface. The operator enters estimated number of people of each type observed at a location (*a*), clicks on the left-hand map to show the location of the observation (*b*), and observes regions of high likelihood of violence on the right-hand map (*c*). In the prototype, *SME* is entered through this same interface (*d*), though the framework supports a direct feed from a NLP analysis program.

simple natural language processor that computes the frequency of profanity and other indications of unrest. The higher the frequency of such traffic, the higher our estimate of *SME*.

Observers on the ground enter local observations of crowd density to CAVE via a web or smartphone interface. Fig. 7 shows the web interface, and Fig. 8 the smartphone interface. The smartphone's geolocation capability provides the location of the observation automatically, allowing police and other observers to update location estimates easily from the ground, and its display of violence estimates provides them with immediate awareness of likely trouble locations.

In the May 20 demonstration, SME estimates were entered by hand, based on manual monitoring of the Twitter feed. The smartphone interface was not deployed to observers on the ground, so crowd estimates were entered through the web interface based on real-time police reports and several streaming video feeds of the event recorded by protestors and journalists among the crowd.

4 Experience with the Model

The CAVE prototype shows the feasibility of integrating real-time data with an agent-based crowd simulation. In the nature of the case, detailed assessment of CAVE's accuracy is not possible, but the Cook County Sheriff's Department commented on the contribution of NEK's tool suite, "The intelligence we received from NEK was relayed to various law enforcement entities, such as the FBI, during the NATO event. The agencies were very appreciative of the information and it helped to enhance all of the intelligence information" [18].

Though the objective of our model is to integrate and present real-world information rather than to study crowd theory, its emergent behavior does provide interesting evidence for the impact of social media. Fig. 9

Fig. 8. Smartphone interface. The smartphone reports its location automatically.

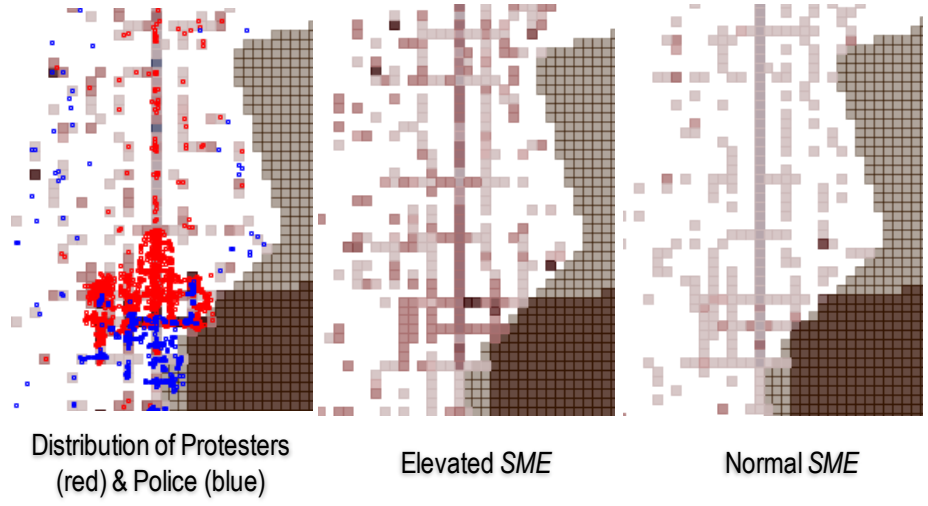

Fig. 9. Dependence of violence on *SME*

shows distribution of the violence estimate for the same distribution of protesters and police, but in two different conditions. High *SME* (middle map) leads to numerous regions of elevated risk of violence, but with low *SME* (right map), only one location near the exclusion zone anticipates high violence.

5 Conclusion

Computational Social Science models have reached a level of maturity that allows them to be used in practical applications. Many such applications, such as crowd monitoring, require the simulation to be continually updated on the basis of real-time information from the domain. CAVE demonstrates how a stigmergic agent-based simulation with apoptotic agents can achieve this objective.

References

1. Bert, F., Podesta, G., Rovere, S., North, M., Menendez, A., Laciana, C., Macal, C., Weber, E., Sydelko, P.: Agent-based Modeling of Land Rental Markets:Comparison between Simulated and Observed Prices in the Argentina Pampas. the Computational Social Science Society of the Americas (CSSSA 2011), Santa Fe, NM (2011)
2. Chao, W.M., Li, T.Y.: Simulating Riot for Virtual Crowds with a Social Communication Model. International Conference on Computer and Computational Intelligence (ICCCI 2011), vol. LNCS 6922, pages 419–427, Springer, 2011)
3. Gilbert, N., Troitzsch, K.G.: Simulation for the Social Scientist. Buckingham, United Kingdom, Open University Press (1999)
4. Gonzalez-Bailon, S., Borge-Holthoefer, J., Rivero, A., Moreno, Y.: The Dynamics of Protest Recruitment through an Online Network. Scientific Reports, 1(197) (2011)
5. Grassé, P.-P.: La Reconstruction du nid et les Coordinations Inter-Individuelles chez *Bellicositermes Natalensis et Cubitermes sp.* La théorie de la Stigmergie: Essai d'interprétation du Comportement des Termites Constructeurs. Insectes Sociaux, 6:41-84 (1959)
6. Grimm, V., Berger, U., Bastiansen, F., Eliassen, S., Ginot, V., Giske, J., Goss-Custard, J., Grand, T., Heinz, S.K., Huse, G., Huth, A., Jepsen, J.U., Jørgensen, C., Mooij, W.M., Müller, B., Pe'er, G., Piou, C., Railsback, S.F., Robbins, A.M., Robbins, M.M., Rossmanith, E., Rüger, N., Strand, E., Souissi, S., Stillman, R.A., Vabø, R., Visser, U., DeAngelis, D.L.: A standard protocol for describing individual-based and agent-based models. Ecological Modelling, 198:115–126 (2006)
7. Grimm, V., Berger, U., DeAngelis, D.L., Polhill, J.G., Giske, J., Railsback, S.F.: The ODD protocol: A review and first update. Ecological Modelling, 221(23):2760–2768 (2010)

8. Jager, W., Popping, R., van de Sande, H.: Clustering and Fighting in Two-party Crowds: Simulating the Approach-avoidance Conflict. Journal of Artificial Societies and Social Simulation, 4(3) (2001)
9. Kantz, H., Schreiber, T.: Nonlinear Time Series Analysis. Cambridge, UK, Cambridge University Press (1997)
10. Parunak, H.V.D.: 'Go to the Ant': Engineering Principles from Natural Agent Systems. Annals of Operations Research, 75:69-101 (1997)
11. Parunak, H.V.D.: Real-Time Agent Characterization and Prediction. International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS'07), Industrial Track, pages 1421-1428, ACM, Honolulu, Hawaii (2007)
12. Parunak, H.V.D., Belding, T.C., Brueckner, S.A.: Prediction Horizons in Agent Models. In Weyns, D., Brueckner, S., Demazeau, Y., editors, Engineering Environment-Mediated Multiagent Systems (Satellite Conference at ECCS 2007), vol. LNCS 5049, pages 88-102, Springer, Dresden, Germany (2008)
13. Parunak, H.V.D., Brueckner, S.: Concurrent Modeling of Alternative Worlds with Polyagents. In Proceedings of the Seventh International Workshop on Multi-Agent-Based Simulation (MABS06, at AAMAS06), pp. 128-141, Springer (2006)
14. Parunak, H.V.D., Savit, R., Riolo, R.L.: Agent-Based Modeling vs. Equation-Based Modeling: A Case Study and Users' Guide. In Gilbert, N., Conte, R., Sichman, J.S. (eds.) Multi-Agent Systems and Agent-Based Simulation, First International Workshop, LNCS, vol. 1534, pp. 10-25. Springer, Berlin, Germany (1998)
15. Reicher, S.D.: The St. Pauls' riot: an explanation of the limits of crowd action in terms of a social identity model. European Journal of Social Psychology, 14:1-21 (1984)
16. Stott, C., Hutchison, P., Drury, J.: 'Hooligans' abroad? Inter-group dynamics, social identity and participation in collective 'disorder' at the 1998 World Cup Finals. British Journal of Social Psychology, 40:359–384 (2001)
17. Strogatz, S.H.: Nonlinear Dynamics and Chaos: with Applications to Physics, Biology, Chemistry, and Engineering. Reading, MA, Addison-Wesley (1994)
18. Towne, B.: Letter from Cook County Sheriff's Dept. to NEK. Letter to Porterfield, T., 23 May (2012).
19. Wasik, B.: #Riot: Self-Organized, Hyper-Networked Revolts—Coming to a City Near You. (2011). http://www.wired.com/magazine/2011/12/ff_riots/
20. Wijermans, N., Jorna, R., Jager, W., van Vliet, T.: Modelling Crowd dynamics: Influence factors related to the probability of a riot. In Proceedings of The Fourth European Social Simulation Association Conference (ESSA 2007), (2007)
21. Zeitz, K.M., Tan, H.M., J.Zeitz, C.: Crowd Behavior at Mass Gatherings: A Literature Review. Prehospital and Disaster Medicine, 24(1):32-38 (2009)

6 ODD Protocol for CAVE

The following description of the CAVE model follows the ODD (Overview, Design concepts, Details) protocol [6,7].

6.1 Purpose

The purpose of CAVE is to provide a real-time window into crowd dynamics that is continuously updated on the basis of real-world information and can forecast (over a limited horizon) how the crowd may behave (in terms of its location, density, and state of agitation). The model was developed to monitor protester dynamics at a protest scheduled along a specific route at the NATO summit in Chicago, IL, May 20, 2012.

6.2 Entities, State Variables, and Scale

CAVE has five types of agents representing humans ("human agents"), a class of agents that manages synchronization of agent distribution with the real world ("location handlers"), and a cellular environment with static geo-spatial information. It processes two kinds of input from the real world (discussed in Section 6.4 "Sensing"): a stream of social media generated by protesters, and estimates of crowd location and size generated by observers

Human Agents.
Human agents are divided into two types, Protesters and Police. Each software agent represents a small number of humans (10 in the experiments reported here). All agents have a Disrespect state variable in [0, 10] which determines whether they engage in violence toward the opposite species. Updating of Disrespect is discussed in 6.3.

Protesters have three subtypes, whose default parameters capture the following intuitions:

- Anarchists are actively trying to make trouble. The initial number is manually configured. They start with Disrespect of 3 and update every time step.
- Followers will not lead in stimulating unrest, but will engage if a fight breaks out.
- Pacifists are reluctant to enter into violence.

Police have two subtypes, whose default parameters capture the following intuitions:

- Riot police have experience, special training, and special equipment for handling disruptive crowds.

Table 1. CAVE Agent Types and Initial Local Variables

Agent Type	Num Agents	Initial *D*	Max Step (m)	*F* (6.3)	*S* (6.5)	*SME* Function Params (6.7)			
						maxCalm	*maxIncite*	*alpha*	*zero*
Anar-chists	10% of protesters	3	15	1.0	5	-0.2	0.8	5.0	0.2
Followers	2x # of Anar-chists	1	10	0.7	15	-0.5	0.5	5.0	0.2
Pacifists	Every-one else	0	5	0.3	50	-0.8	0.2	5.0	0.2
Riot Police	84% of Police size	0	10	1.2	See Sec-tion 6.5	Not Applicable			
Patrol Officers	16% of Police size	2	15	1.5					

- Patrol officers have less extensive training and do not have special equipment.

The user initially specifies the total number of agents of each type (in this case, 30,000 protesters and 12,500 police). Table 1 describes the three types of protester agents and two types of police agents in terms of their percentage of the population, their initial level of disrespect *D* at the start of the run, the maximum distance they can move in a time step (Section 6.3), and parameters governing their response to a nearby fight (Section 6.3), initial distribution (Section 6.5) and (in the case of protesters) their response to *SME* (Section 6.7).

Environment.

The environment is a square lattice with cells 40 m on a side. Cells are labeled as road or non-road, based on a map of the urban area. They are also labeled as security area (from which all agents are excluded) and non-security. Road cells have two subtypes: the designated protest route, and other roads. Agents are located on a continuous scale, and thus may be at different locations within a cell.

A global variable, accessible to all agents, records Social Media Energy (*SME*), a measure of protester unrest derived real-time from tweets among protesters.

Location Handlers.

Each time an observer reports a crowd of protesters at a location, CAVE instantiates a location handler at the cell nearest to the location report that is marked as a road. This handler seeks to align the population of agents at that location with the reported observation, using the apoptosis of human agents (discussed in Section 6.3). The impact of a location handler decays as the observation it represents ages, and the handler dies after a fixed horizon (60 wall-clock minutes in this implementation).

6.3 Process overview and scheduling

Agents representing humans have two processing loops, a short "behavioral loop" that executes once each simulation cycle (roughly, once per minute of simulated time), and a longer "lifetime loop" that governs the agent's synchronization with the real world. The agents interact with one another via digital pheromones that they deposit in the environment, and the environment actively manages these pheromones.

Human Agent Behavioral Loop.
Agents execute in random order without replacement (no agent can execute twice until all agents have executed once).

An agent carries out the following steps each time it executes.

- If it is on a road cell, it deposits several pheromones reflecting its location and current state. The pheromone flavors are:
    - Position, a unit deposit labeled with agent type
    - Disrespect, equal to the agent's current Disrespect level, labeled with its type
    - Fight, a unit deposit if the agent is in a fight
    - FightResolved, a unit deposit if the agent has finished fighting.
- It scans its cell for pheromone levels.
- It updates its level of disrespect, based on
    - What it observes in its environment and
    - If it is a protestor, the level of *SME* (per Section 6.7).
- It acts based on its level of disrespect (Table 3)

Table 2 shows the events that change the Disrespect level of each type of agent.

Table 2. Events Changing Disrespect

Type of Agent	Action That Causes Disrespect	Change in Disrespect
Protester or Police	Fight takes place next to agent	Normalized disrespect pheromone * *F* (Table 1)
Protester	Social Media Energy	Per function discussed in Section 6.7
Police	Fight resolved next to agent	-4

An agent's action depends on its level of disrespect, as indicated in Table 3:

Table 3. Behavior Triggered by Disrespect Level

Resulting Behavior	Triggering Level of Disrespect	
	Protester	Police
Go towards own type	< 5	< 7
Go towards other type	> 5	> 7
Fight	≥ 8	≥ 9

When an agent decides to move, it senses the level of presence pheromone of the target agent type (per Table 3) in each cell of its Moore neighborhood. It computes the vector sum of nine vectors, all originating at its location (which need not be at the center of a cell). Each vector points toward the center of one cell in the Moore neigh-

borhood, and its length is proportional to the product of the presence pheromone in that cell, and the distance between the agent's current location and the center of the cell. To break symmetries, the agent then adds a randomly oriented vector of length .25 to the vector sum. Then the agent takes a step in the direction of the final vector. If the vector is longer than the agent's maximum step length (from Table 1), the agent takes a step of maximum length. Otherwise it takes a step of length equal to the vector.

When an agent decides to fight, the fight lasts for a time uniformly selected between 5 and 15 cycles. During this time, the agent makes takes no other actions. At the end of the time, if the agent is a protester, it ends its lifetime (simulating arrest) and is reborn.

Human Agent Lifetime Loop.

- Agents are apopotic, which means that they live for a fixed period of time (uniformly sampled for each agent from the range [50, 150]), then die. When one agent dies, it is replaced by another one of the same type and resamples its subtype according to Table 1, at a location influenced by observers' reports of crowd population, and sets its initial level of disrespect to the average of other agents of its species in its new location. If that average is higher than the default for its subtype, the average is attenuated by 0.9 (for a protester) or 0.8 (for police).

When an agent dies, it is reborn as an agent of the same type and subtype, at the center of a cell selected in the following way.

- The reborn agent first seeks for location handlers that are requesting an agent of its type (and in the case of Anarchists, its subtype). If one or more location handlers is requesting an agent of this subtype, they bid for the agent based on their current need for agents and the age of the observation that they represent.
- Otherwise the agent is distributed following the initialization protocol (Section 6.5). If the assigned cell is occupied by a location handler whose reported value is lower than the current population of the cell, the assignment is rejected and a new destination cell is computed.

Environmental Dynamics.
At each time step, the environment evaporates all digital pheromones in each cell by multiplying them by 0.95. Local pheromone concentrations that have fallen below 0.1 are set to 0, reducing the computational effort in maintaining the fields.

6.4 Design concepts

Basic principles.
The model is based on the Elaborated Social Identity Model, inspired by [8]. The contribution of this paper is not in extending this model in novel ways, but rather in

demonstrating how it can be coupled with real-time data to provide short-term forecasts of crowd unrest and likely hotspots.

Emergence.
The primary result of interest is the location of hot spots, where the aggregate level of Disrespect is high and where violence is likely to break out as a result. Such a hotspot cannot be generated by any single agent, but is the emergent result of many agents participating locally in the feedback loops shown in Fig. 4.

Adaptation.
Agents adapt in two ways, as shown in the bottom loop of Fig. 4. They modulate their own level of Disrespect based on the state of their environment, then they select their next action based on their level of Disrespect.

Objectives.
The agents in CAVE have no individual objectives.

Learning.
CAVE agents have only very primitive learning, consisting of variable increment or decrement of their level of Disrespect (depending on environmental circumstances), coupled with a continuous exponential decrease (forgetting) of Disrespect.

Prediction.
CAVE agents do not individually predict the future. They simply respond directly to environmental stimuli, but because the simulation clock runs faster than real time, they are living in the near future, so that their collective state represents a prediction.

Sensing.
CAVE agents have access to two kinds of variables: endogenous variables generated by the agents themselves, and exogenous variables calibrated to information from the real world.

There are two exogenous variables.

1. The system as a whole maintains the global Social Media Energy (*SME*) variable based on social media generated by the protesters and monitored by the CAVE users.
2. Observers can report the location and size of crowds of protesters and the percentage of Anarchists in such a crowd.

Endogenous variables include the density and disrespect of each type of agent (based on pheromones that they deposit), and the presence of fights, in their cell.

Interaction.

All agent interactions are indirect, mediated through the environment (Fig. 4) via pheromones representing agent presence, fights, and levels of Disrespect.

Stochasticity.
CAVE uses stochasticity in the following decisions:

- Lifetime of each human agent
- Duration of a fight
- Spatial and subtype distribution of agents (initially, and when they are reborn after apoptosis)
- Agent movement (addition of randomly oriented 0.25 vector to movement vector)

When a new agent is born, its location and subtype is selected stochastically, weighted by the overall observed distribution of crowd members as reported by observers.

Collectives.
All agents have species and subtypes as outlined in Section 6.2, but they act individually. Each agent represents ten humans, but there are no individual representations of those humans.

Observation.
CAVE delivers its results through a graphical display (Fig. 7) that shows likely hotspots based on a violence estimation formula. The violence level reported through the display is linear in the mean Disrespect across Protesters in a cell, ranging from 0 if the mean Disrespect is 0 to 100% if the mean Disrespect is equal to or greater than the Protesters' fight threshold (0.8 in this implementation, Table 3).

6.5 Initialization

CAVE is initialized with human agents based on the input number of protesters and police. The proportion of each subtype of agent is based on Table 1. These agents are then allocated probabilistically to the cells in the area of interest. Each cell is assigned a weight of receiving an agent of a given subtype. Then we normalize the weights so that they total 1 and assign each agent to a cell chosen with probability equal to its normalized weight. We assign weights using the following protocol.

Cells within the security areas, or not on a road, have weight 0.

For each cell, we compute the Euclidean distance between its center and the center of the nearest protest route cell, normalized to [0, 1].

The weight that a cell will receive a protester depends on this normalized distance d and the S parameter for the protester subtype (Table 1), in a way that makes pacifists more likely to stay close to the route, and followers and anarchists increasingly likely to stray from the route. The function is:

$$w(protestor) = 1 - \frac{e^{-d\,S} - 1}{e^{-S} - 1}$$

Fig. 10 illustrates this function for followers (S = 15).

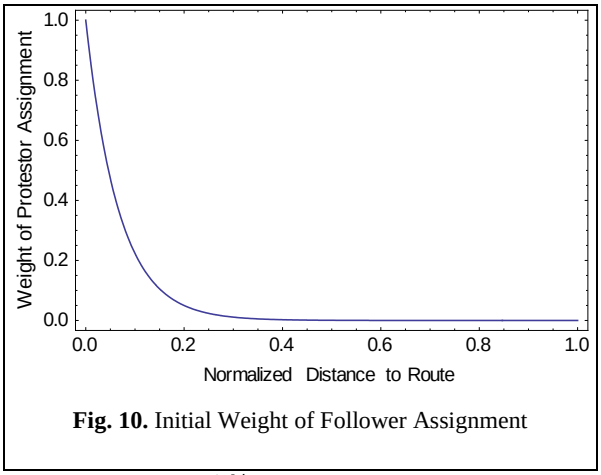

Fig. 10. Initial Weight of Follower Assignment

The distribution function for police is different, to capture the intuition that they should be near but not on the protest route. We distinguish two cases, depending on whether the cell's normalized distance to the route is less than or greater than 0.1:

$$w(police) = \begin{cases} d < 0.1: \dfrac{e^{5\,d/0.1} - 1}{e^5 - 1} \\ d \geq 0.1: 1 - \dfrac{e^{-2\,(d-0.1)/0.1} - 1}{e^{-2\,(1-0.1)/0.1} - 1} \end{cases}$$

Fig. 11 plots this function.

6.6 Input data

CAVE is initialized with the static geospatial terrain and estimates of overall Protestor and Police population size. It continuously draws data from two sources: an estimate of Social Media Energy from Twitter feeds monitoring known anarchists and hashtags related to the protests and heavily used by the protestors, and observations of crowd concentration.

6.7 Submodels

At several points in this protocol, we transform one variable into another through a function of the form

$$y(x) = \frac{e^{\alpha x} - 1}{e^{\alpha} - 1}$$

In spite of its superficial complexity, this function is simply a way to modulate a monotone curve from (0, 0) to (1, 1) from a straight line

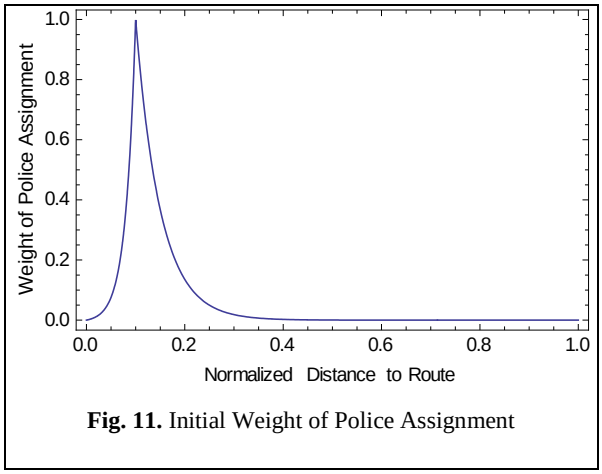

Fig. 11. Initial Weight of Police Assignment

(when α is very small, say 0.01) to an increasingly steep convex curve (for high α). Varieties of this function let us model decreasing instead of increasing functions, and introduce concave instead of convex deviations from linearity.

In addition to the uses of this form that have already appeared, it is used in the function that converts *SME* into a change in a protester's Disrespect level. The function has four parameters, which differ across subtypes of protester (Table 1):

- *maxCalm* is the largest reduction in Disrespect, achieved when *SME* = 0
- *maxIncite* is the largest increase in Disrespect, when *SME* = 1
- *zero* is the value of *SME* at which there is no change in Disrespect
- *alpha* is a shaping parameter, which increases sigmoid shape of the curve.

The function is

$$D = \begin{cases} SME < zero: maxCalm \dfrac{e^{alpha\ (zero - SME)/zero} - 1}{e^{alpha} - 1} \\ SME \geq zero: maxIncite \dfrac{e^{alpha\ (SME - zero)/(1 - zero)} - 1}{e^{alpha} - 1} \end{cases}$$

Fig. 12 shows the curve for an Anarchist agent.

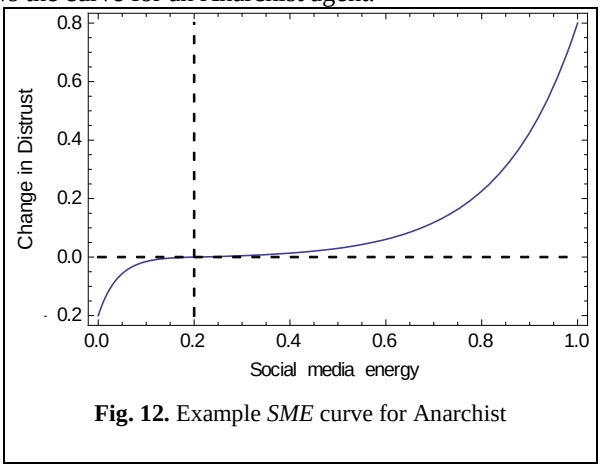

Fig. 12. Example *SME* curve for Anarchist