Two Visualization Tools for Analyzing Agent-Based Simulations in Political Science

R. Jordan Crouser and Daniel E. Kee ■ *Tufts University*

Dong Hyun Jeong ■ *University of the District of Columbia*

Remco Chang ■ *Tufts University*

Research in the social and behavioral sciences often uses agent-based modeling (ABM) to simulate and explore behaviors such as collaboration,[1] conflict,[2] violence,[3] and population change.[4] Researchers have also used agent-based models to identify a country's political patterns, which might indicate the imminence of civil unrest and help predict catastrophic events.[5] ABM represents a behavioral system as a collection of autonomous entities or agents. Each agent interacts with other agents according to a set of rules and goals, and over time it might influence and be influenced by the agents around it.

As increased computing power becomes more widely available, scientists can simulate more complex systems, which in turn generate substantially larger datasets that must then be analyzed and interpreted. Unfortunately, the methods and tools available to social scientists for analyzing simulation results can't support datasets of such magnitude, making it difficult for scientists to effectively interpret and analyze the results.[6]

In addition, ABM is a stochastic simulation technique, using small random perturbations to the interaction rules and running each simulation hundreds or even thousands of times to avoid local minima and to generate a distribution of sample behavioral patterns. Analysts must therefore be able to compare simulated behaviors between and across distinct runs and to piece together many simulation runs into a single, cohesive overview.

To help address these challenges, we collaborated with domain experts to develop two visual analytics tools to support analysis of agent-based models in political science. With MDSVis (Multidimensional Scaling Visualization), analysts can explore the simulation space, finding similar patterns at an aggregated level and finding the dominant factors affecting agent behavior. With SocialVis, analysts can focus on one simulation run, exploring relationships between time steps and geographic regions.

MDSVis (Multidimensional Scaling Visualization) lets analysts explore the simulation space of agent-based models, finding similar patterns at an aggregated level and finding the dominant factors affecting agent behavior. SocialVis lets analysts narrow their analysis to a single simulation run, exploring relationships between time steps and geographic regions.

Domain Characterization

Through behavioral-simulation analysis, scientists seek to uncover the sociopolitical and socioeconomic forces that control and influence group behaviors and to predict behavioral patterns using real-world data. A better understanding of how these forces influence group behavior and the ability to make more accurate predictions can greatly influence how we view real-world behavioral systems and better inform decisions regarding domestic stability, foreign policy, and more.

The first step in this process is constructing an

accurate model. Researchers develop existing political theories based on observed behaviors and interactions into an agent-based model. They then seed the model with data collected in the field about political party affiliation, level of violence, protests, regional and local conflicts, and more.[7] Using this input, the agent-based model produces a large amount of data representing a distribution of possible behavioral patterns over a period of time. Analysts then study these results to try to extract a cohesive story explaining the relevant interactions and to identify interesting or highly likely outcomes.

We collaborated with domain experts to develop two visual analytics tools to support analysis of agent-based models in political science.

Although statistical analysis of the resulting data is possible, it often proves insufficient. Because of these simulations' complexity, the generated datasets require expert analysis to interpret the results as valid behavioral patterns and fully understand the forces controlling the interactions observed in the simulations. The amount of data is so large that examining it by hand would require countless hours, and so the data must often be simplified and some of the subtlety sacrificed to conserve time and energy.

Clearly, computational support for these analytical processes is critical to social and political science. Systems for supporting expert analysis must present the data in a manner that preserves information and allows for deep, low-level exploration while maintaining a manageable overview.

Design Considerations

Our early conversations with our collaborators indicated that their analysis of agent-based simulation data faces three main challenges: exploring the data as a whole to generate initial hypotheses at both the global and single-simulation level, comparing simulation runs, and incorporating domain expertise into data analysis.

Supporting Hypothesis Generation and Exploration

To support analysis, we must first enable analysts to view the data in a meaningful way. At the overview level, this means giving analysts intuitive mechanisms for examining the aggregated data, comparing high-dimensional simulation runs, and identifying trends and outliers for further exploration. At the detail level, we must provide an organized mechanism for drilling down to a single run, enabling analysts to explore the behaviors of a single set of conditions. We must also provide a useful tool for debugging the simulation.

Currently, analysts use line graphs and statistical plots of each dimension to make comparisons, and they compare the individual variables' values to drill down to a single run. This process is laborious, highly error prone, and fails to provide a sense of how the dimensions interact.

Comparing Simulation Runs

As our collaborators indicated, comparing distinct simulation runs is often useful. For example, analysts might want to explore outliers to determine whether they represent legitimate but unlikely outcomes or are simply noise. Until now, no technology has existed that helps analysts do this. To compare simulation runs, analysts have had to compare each variable's values independently, leaving them without a holistic overview of the similarities and differences between the runs.

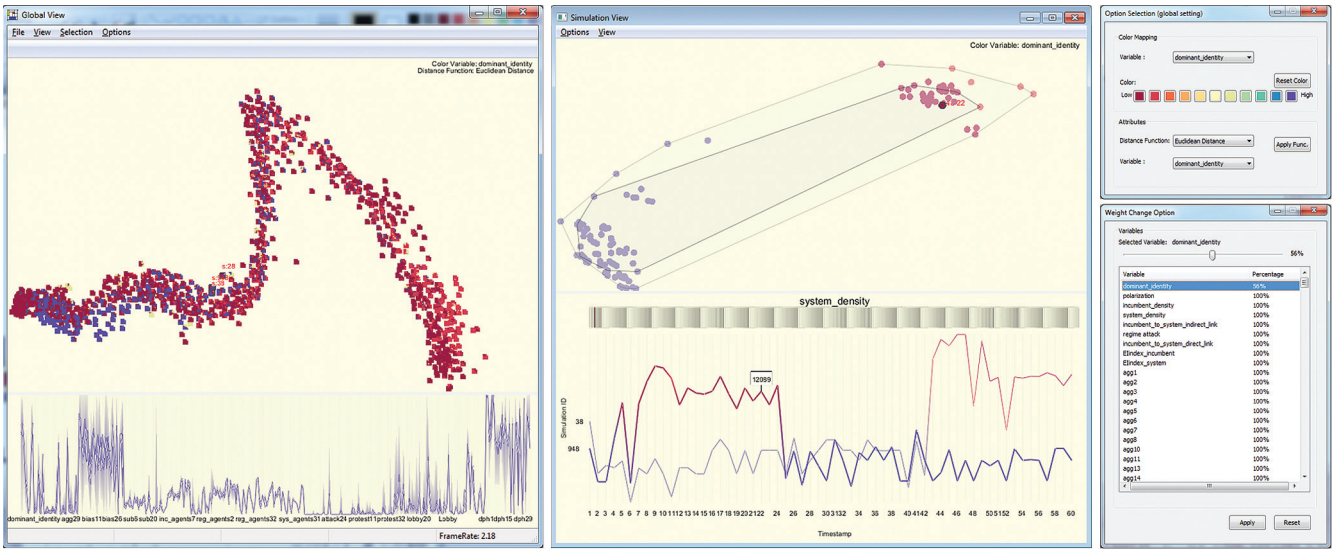
Leveraging Domain Expertise

As the result of early offline explorations of data, Ian Lustick and his colleagues at Lustick Consulting developed the Dynamic Political Hierarchy (DPH), a model that combines theories of cross-cutting cleavages, nested institutions, and dynamic loyalties.[5] They found this model incredibly useful for political forecasting. However, the ability to observe changes in DPH structures over time was limited and involved the laborious process of generating and comparing diagrams one at a time.

Our Visual Analytics Tools

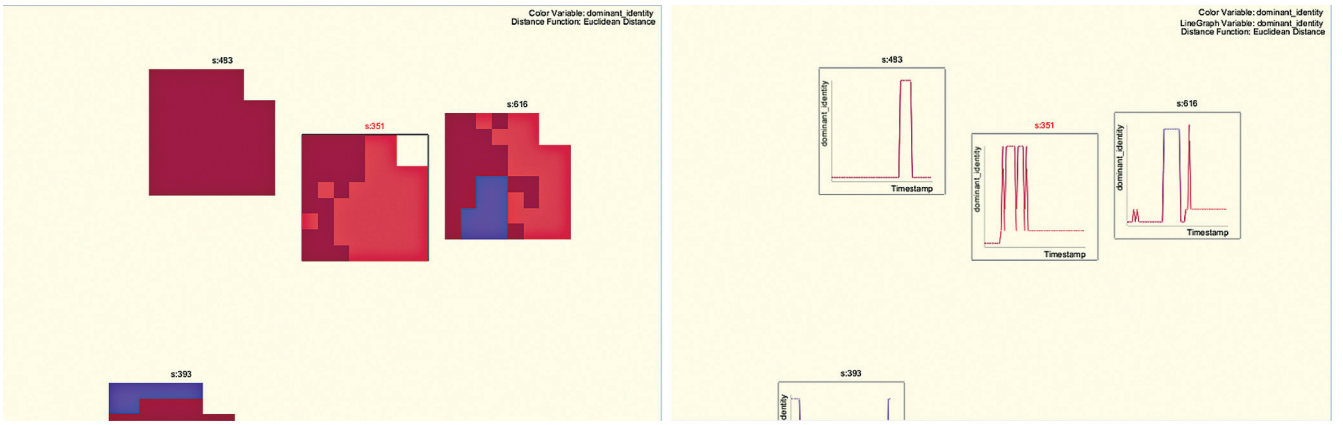
To address these challenges, we designed and implemented MDSVis and SocialVis. To support the analysis of complex political simulations, both tools employ a *coordinated multiple views* (CMV) architecture, which lets analysts customize the views to suit their analytical process. In this architecture, interaction with one view is immediately reflected to all the other views.

To effectively coordinate each view, we implemented an interaction manager that handles all keyboard and mouse interactions. In addition, the selection operation in all views and a zoom mechanism in some views help analysts focus on specific simulations or time steps. Because the two tools are closely connected and maintain a consistent vi-



(a) (b) (c)

Figure 1. The MDSVis (Multidimensional Scaling Visualization) interface. (a) The global view shows all the simulations. In that view, the projection view (top) shows the relationship between simulations based on a given distance function, and the data view (bottom) displays the simulations' attributes. (b) The simulation view presents the simulations selected in the global view. In that view, the cluster view (top) displays the selected simulations' similarity, and a temporal view (bottom) displays those simulations' state changes. (c) Two control panels let analysts modify variables and colors (top) and control each dimension's contribution to the MDS calculation (bottom).



(a) (b)

Figure 2. In the projection view, analysts can switch the glyph representation while navigating the projection space. (a) These pixel-oriented glyphs display 60 time steps of each simulation by following Hilbert-curve ordering. (b) These line graphs represent the temporal changes for a variable.

sual metaphor whenever appropriate, analysts can easily transition from analyzing multiple simulations with MDSVis to closely exploring a selected simulation with SocialVis.

MDSVis

This tool consists of four views and two control panels (see Figure 1).

The projection view. MDSVis represents simulations by applying a distance function and multidimensional scaling (MDS) in the projection view. Because limitations exist on applying MDS directly to large-scale input data, we perform analysis of variance. We compute the mean variance of all the time steps to determine the variable distribution's center for each simulation, and then we apply the distance function. Although finding a semantically meaningful distance function is important, identifying the appropriate contribution of all variables requires significant computational time. We use a simple Euclidean distance function and let analysts manually control each dimension's contribution. We then apply MDS to reduce the simulations' dimensionality. By default, MDS runs 1,000 iterations. The top of Figure 1a shows all 1,000 simulations.

The projection view displays each simulation as

a pixel-oriented glyph (see Figure 1a, top). We do this by employing Hilbert-curve ordering to arrange each time step with color information. This technique provides continuous curves while maintaining good locality of information. For mapping each time step, we set the Hilbert curve order to 8, which covers up to 8 × 8 sizes. We then use color coding to represent the selected variable at each time step. Analysts can select this parameter in the appropriate control panel (see Figure 1c, top) or can switch to a line graph representation (see Figure 2).

Because MDSVis maps each subregion of the parallel-coordinates visualization to a variable, analysts can select a variable by simply choosing a subregion.

The data view. Each simulation is controlled by 351 variables (attributes). To present them, we use a parallel-coordinates visualization. Although understanding 1,000 simulations with 351 variables through such a visualization can prove difficult because of clutter, this visualization technique is useful when the data exhibit patterns or underlying structure.

In the parallel-coordinates visualization, MDSVis selects a color attribute by referencing each simulation's political structure. Most variables are mapped by the DPH, which characterizes a country's political structure on the basis of the relationships and strengths of individual political, racial, ideological, and religious groups. So, a frequency analysis counts the political structures to determine the dominant structure in each simulation. For instance, if approximately 60 percent of the dominant identities in a simulation fall in a given category, MDSVis will select that category as the representative political structure. MDSVis will then use the corresponding color attribute to represent the simulation as a line graph.

In the data view, each line denotes one simulation. When an analyst highlights or selects simulations in the projection view, MDSVis emphasizes those simulations in the data view by hiding all the other simulations in the parallel-coordinates visualization. It also displays the highlighted simulations' mean variance using gradient color mapping (see Figure 1a, bottom). With this feature, analysts can intuitively identify the variance over the 60 time steps in each simulation.

The cluster view. Once analysts have identified and selected interesting simulations in the projection view, the cluster view represents those simulations' time steps. MDSVis maps each time step to a unique circle in this view (see Figure 1b, top).

As with the projection view, we apply MDS to reduce dimensionality. In this case, we apply MDS to all the time steps in the selected simulations. Because the cluster view treats each of the 60 time steps as a data element, MDSVis will compute similarities among 120 data elements when an analyst selects two simulations.

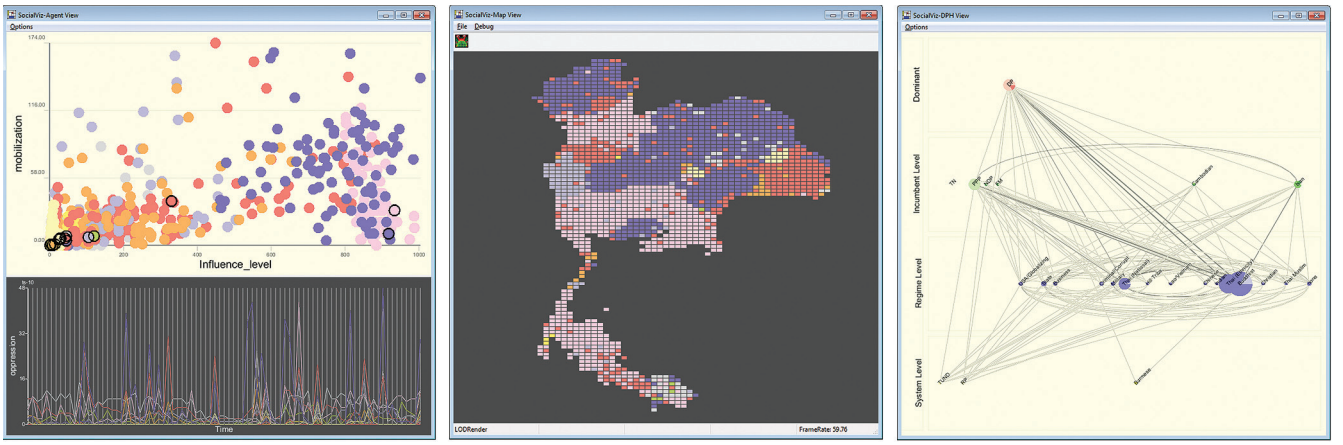
When analysts select more than two simulations, representing all the time steps in the projection view will make it difficult to determine which simulation produced each time step. To avoid this ambiguity, MDSVis computes a convex hull to form a group boundary around each simulation (see Figure 1b, top). Each convex hull indicates a cluster of each selected simulation. If the analyst highlights an item (that is, a time step) by hovering over it, the cluster view highlights the corresponding simulation's convex hull.

The temporal view. This view displays all attributes related to each time step in a parallel-coordinates visualization. As the bottom of Figure 1b shows, the layout has two components: a variable selector is above the parallel-coordinates visualization.

Because MDSVis maps each subregion of the parallel-coordinates visualization to a variable, analysts can select a variable by simply choosing a subregion. Or, they can select a variable from the appropriate control panel (see Figure 1c, bottom). On the basis of the selection, MDSVis displays the corresponding information in the parallel-coordinates visualization. This visualization indicates time steps intuitively along the *x*-axis. As Figure 1b shows, MDSVis uses the color attributes from the cluster view when rendering lines in the parallel-coordinates visualization. From this, analysts can identify what factors change DPH structures.

Control panels. The two control panels let analysts manage the visualization's input parameters. The first lets analysts modify visualization attributes (see Figure 1c, top). Analysts can change variables and modify color mappings. Because MDSVis creates the color mapping by referencing the selected variable, whenever analysts select a different variable, the visualization will display the corresponding information.

The other panel controls a variable's contribution to the MDS calculation (see Figure 1c, bottom). A



(a) (b) (c)

Figure 3. The SocialVis tool's four views. (a) The bubble chart view (top) shows the correlation between mobilization and the influence level for each group of agents. The temporal view (bottom) represents each group's activities over time. (b) The map view shows the agents' geographical distribution. (c) The Dynamic Political Hierarchy (DPH) view assigns each group of agents a level in the hierarchy: dominant, incumbent, regime, system, or antisystem.

contribution change from 100 to 50 percent indicates that the selected variable's weight has changed to 0.5. When the contribution is 0 percent, MDSVis won't use that variable to compute similarity.

SocialVis

Figure 3 shows the tool's four views.

Bubble chart view. This view (see Figure 3a, top) lets analysts examine the actions and interactions of each agent or political group by comparing the correlation between its controlling variables. Analysts select variables to compare through a control panel. If two variables maintain a positive correlation, the dot pattern will slope from the lower left to the upper right. SocialVis determines the color attribute by the identity of each agent or group, employing the same encoding metaphor used in MDSVis.

Temporal view. This view (see Figure 3a, bottom) represents the activities of each agent of a political group over time as a line, with the line's color matching the group's color. When analysts highlight a line or time dimension, the corresponding information is reflected in all other views.

Map view. This view (see Figure 3b) represents geographical information corresponding to each agent. Because in this case we performed the political simulation on data gathered in Thailand, we use a map of Thailand. Each region is mapped with an agent, whose color corresponds to its group.

DPH view. This view (see Figure 3c) shows the groups of agents and how their relationships impact a political system's structure and stability. Its configuration characterizes a country's political structure on the basis of the relationships and strengths of individual political, racial, ideological, and religious groups.

We assign each group of agents a level in the hierarchy: dominant, incumbent, regime, system, or antisystem. The line between two groups represents the relationship, and its thickness indicates how strongly the groups are connected. By default, SocialVis displays all linkages among groups.

Because the DPH view uses graph drawing, it has that approach's commonly known limitations (cluttering and line crossing). To minimize these limitations, it uses B-spline curves to create curvy lines. Also, it emphasizes only highlighted linkages when analysts interact with groups.

The DPH view represents each group as a pie chart holding two pieces of information: the number of activated identities (agents currently mobilized by the group) and total number of subscribed identities. The darker region indicates the proportion of activated identities.

Usage Scenarios

Our collaborators identified two main goals for analyzing agent-based simulation data. First, they want to identify potential outcomes and estimate their likelihood. Second, they want to identify and analyze outlier runs with high potential impact and determine what caused the simulation to behave differently during these runs.

Identifying Potential Outcomes

We initialized MDSVis with data from the VirThai simulation data created by our collaborators as part of DARPA's Integrated Crisis Early Warning System project.[7] (VirThai is an agent-based simulation of

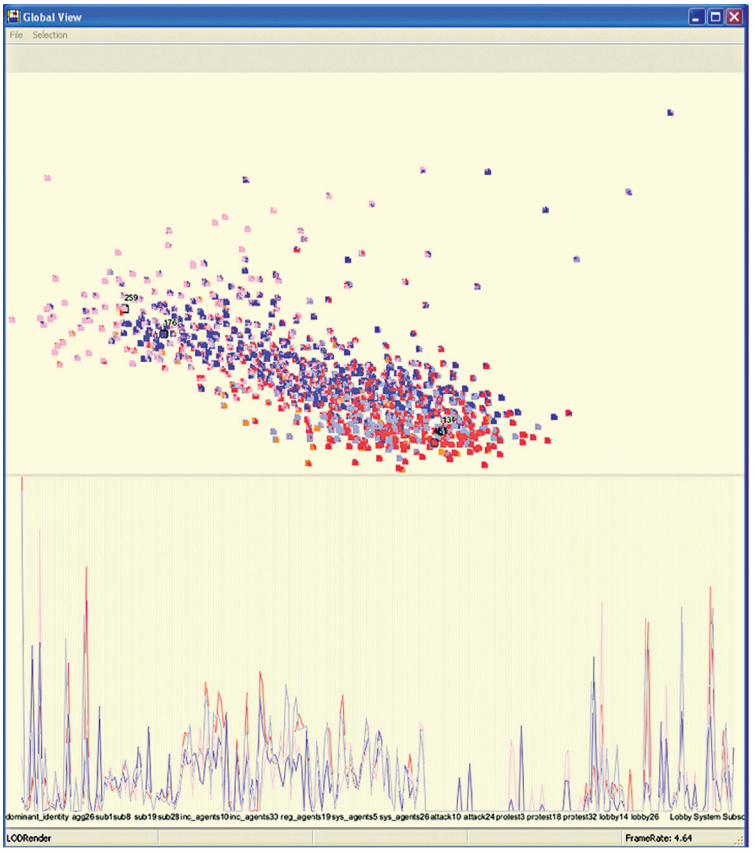


Figure 4. A representation of the data with pixel-oriented glyphs of the dominant-identity attribute in the projection view. Because this attribute contributes a little to the distance function, we can use it as a label for each simulation in this context. Runs that prominently feature the Buddhist (red) or Thai Ethnic (light purple) group as the dominant identity are clustered on the right; runs that prominently feature the Red Shirts (dark purple) or Yellow Shirts (pink) are clustered on the left.

political violence and unrest in Thailand.) First, they represented the data with pixel-oriented glyphs of the dominant-identity attribute in the projection view (see Figure 4) to explore how the simulation runs are clustered and how the clustering correlates to this attribute. Because this attribute contributes a little to the distance function, we can use it as a label for each simulation in this context.

In Figure 4, runs that prominently feature the Buddhist (red) or Thai Ethnic (light purple) group as the dominant identity are clustered on the right; runs that prominently feature the Red Shirts (dark purple) or Yellow Shirts (pink) are clustered on the left. Because the Buddhist/Thai Ethnic clustering is roughly the same size as the Red Shirts/Yellow Shirts clustering, the probability of Thailand's future resembling either of these outcomes is similar.

The analysts then selected one run from each of the dominant identities in the two clusters to see how each run's attributes differ. They looked specifically at the lobby, protest, and attack attributes. As Figure 5 indicates, the two clusters differ significantly for the lobby and protest attributes but not the attack attribute.

The analysts couldn't make strong predictions about Thailand's future from this analysis. However, they hypothesized that runs in the Buddhist/Thai Ethnic clustering would exhibit high levels of lobbying and low levels of protest, whereas runs in the Red Shirts/Yellow Shirts clustering would exhibit the opposite. To confirm their hypothesis, they selected 10 runs from each cluster and observed similar patterns for each attribute (see Figure 6).

Identifying Unlikely but High-Impact Outcomes

For this scenario, the analysts returned to the projection view (see Figure 4). Adding two of the outliers to the subset of runs selected in the previous scenario, they obtained the temporal view of the attack attribute (see Figure 7). In the four runs from the previous scenario, little noticeable difference existed between the levels of attack for the runs. However, the additional outlier runs show several spikes indicating high levels of attack relative to the runs in the clusters.

To further explore why these outliers display such high levels of attack, the analysts switched to SocialVis. First, they used the time series and bubble chart views to confirm the spikes they observed with MDSVis. They then examined the DPH view of the time steps immediately preceding the increase in attacks.

They observed two patterns. First, in the two time steps immediately before the attacks, the Thai Ethnic identity moved from the regime level to the system level (see Figure 8). Second, the Isan ethnic group moved from the system level to the incumbent level (see Figure 8). These patterns occurred immediately before nearly all the spikes in attacks. From this, the analysts leveraged their domain expertise to conclude that, for this run, the high levels of attack probably resulted from the Thai Ethnic group's alienation whenever the Red Shirts aligned closely with the minority Isan group.

Expert Evaluation

We first introduced the analysts to each tool's design and utilities and to the visualizations that each tool supports. Next, we invited them to use the tools for a few days, and then we collected their comments through informal interviews. Overall, the analysts reported that our tools were invaluable and that they met all the design considerations we had collectively identified at our collaboration's onset.

MDSVis

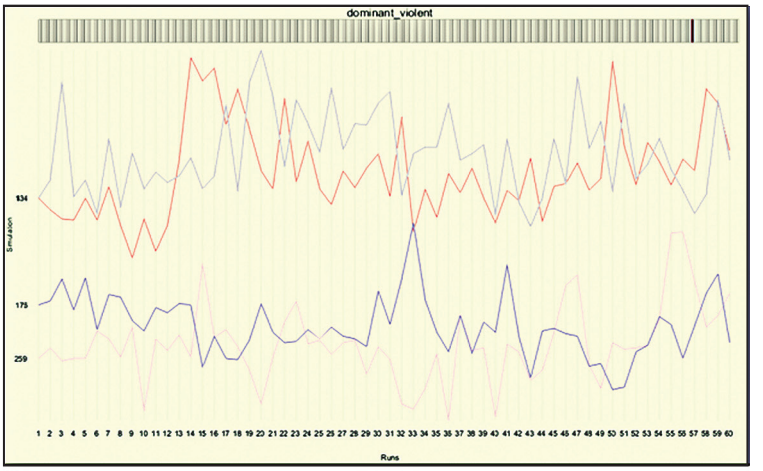
The analysts reported that MDSVis was extremely useful for comparing runs across multiple data dimensions. One analyst mentioned that

> This is the first time we've really been able to group runs according to multidimensional similarity. Until this point we didn't even really have a rudimentary strategy ... and even univariate similarity comparisons relied on comparing [a] large number of time series or comparing means.

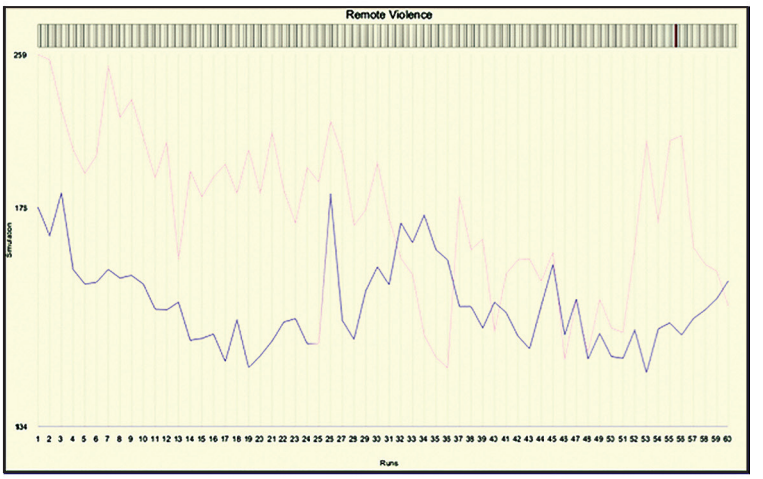
MDSVis broadens the range of possibilities for analysis by providing a straightforward mechanism for performing multivariate clustering on complex data, as well as greatly reducing the computation time for performing traditional comparisons.

The analysts also reported that MDSVis would greatly reduce their analytical process's entry barrier. Although identifying and grouping similar runs and then drilling down into the data to determine what makes those runs unique was previously possible "based on a high level of familiarity with the model ..., the process was often opaque," according to one analyst. Another analyst said that, by using MDSVis to identify groups of similar runs and then employing the parallel-coordinates visualization and time series view to examine the simulation runs' details, "a new user is able to explore a dataset and find interesting relationships, or an experienced user can more quickly understand a new dataset."

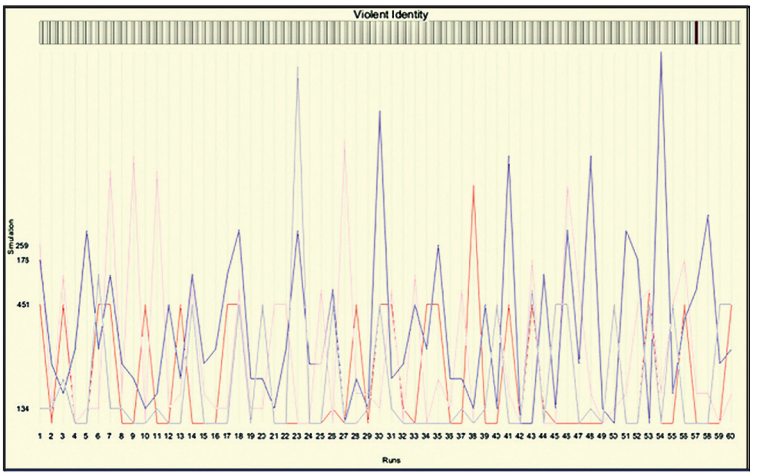
During the evaluation, the experts also identified a few shortcomings of MDSVis. They noted that it wasn't particularly well suited for presentation owing to the challenges in comparing across multivariate space. They also noted that although they found it useful to be able to alter the distance function by using the control panel to modify the variable weights, computation speed could be problematic. Another drawback was that patterns across many dimensions tended to cancel each other out. The analysts suggested that in some cases, displaying patterns among fewer variables might be more intuitive and show stronger relationships. However, in datasets in which relationships are generally weak, comparing across all variables instead of only a subset might help illuminate less obvious patterns.



(a)



(b)



(c)

Figure 5. The analysts selected one run from each of the dominant identities in the two clusters to see how the lobby, protest, and attack attributes differ in each run. The two clusters differ significantly for (a) lobby and (b) protest but not (c) attack.

SocialVis

Analysts agreed that SocialVis provided a much more efficient framework for exploring individual trajectories and different variables. One analyst stated that to accomplish this task, they previously would "have to open the model in PS-I and watch the particular trajectory run or use off-the-shelf software (e.g. Excel, Stata)." SocialVis let analysts straightforwardly access and visualize many of the variables at work in their model.

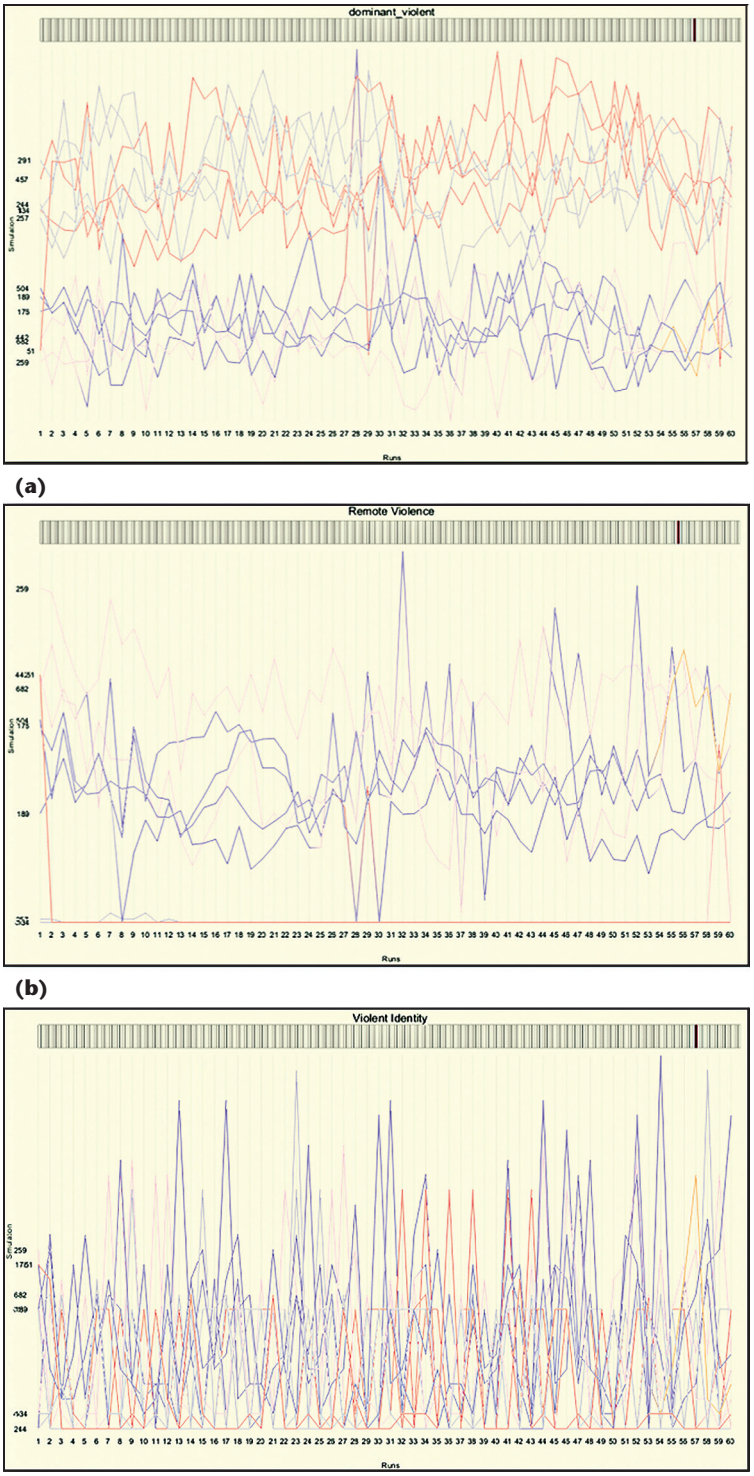


Figure 6. Analysts hypothesized that runs in the Buddhist/Thai Ethnic clustering would exhibit high levels of lobbying and low levels of protest, whereas runs in the Red Shirts/Yellow Shirts clustering would exhibit the opposite. To confirm this, they selected 10 runs from each cluster and observed similar patterns for (a) lobby, (b) protest, and (c) attack.

Another analyst noted that

> One of the great advantages of SocialVis is its speed, which allows a user to analyze the configuration of a landscape over an entire run very quickly without having to flip back and forth between a series of images. Some of the views, like the sequential DPH visualization, were not available to us at all; [before SocialVis] we only had the ability to generate the visualization from individual time steps, which is a very time-intensive process.

The only drawback the analysts noted was that not all variables and attributes in the model were available for viewing, such as the rules and functions operating in the model.

Areas for Improvement

On the basis of the analysts' encouraging feedback, we plan to refine our tools and make them available for use by other social and political scientists. Our refinements are focusing on the following areas.

Identifying Appropriate Distance Functions

Our current implementation uses the Euclidean distance as a proof-of-concept distance function. However, in this distance measure, the attributes aren't normalized and thus have uneven weighting depending on each attribute's range of values. Analysts could compensate for this by adjusting the contribution for an overrepresented or underrepresented attribute in the appropriate MDSVis control panel. However, it would be much more intuitive if the contribution values in the control panel correlated with the contribution of attributes in the distance function. We also want to explore the utility of offering analysts several initial predefined options, depending on the data being examined, to minimize the time and effort to properly tune the distance function.

Another issue is that the Euclidean distance tends to perform poorly when similar features shift slightly in time. This weakness is evident especially with the agent-based simulation data we used in experiments, in which attributes can vary greatly between consecutive time steps. Intuitively, the distance between two runs that are identical with the exception of a slight shift in time should be almost nonexistent. However, the Euclidean distance has no mechanism to recognize this.

So, we've considered several other distance measures. Dynamic time warping[8] can effectively handle temporal shifting but is computationally intensive. Symbolic aggregate approximation[9] can be used to determine a lower bound of the Euclidean distance between two time series in a fraction of the time. So, we could apply it to subsequences of

the time series to quickly find similar features that shift in time.

We want to continue exploring different distance measures to afford analysts better performance and increased control when using these tools. One area of interest is automatic generation of distance functions. We're evaluating the effectiveness of using a computational "best guess" approach coupled with iterative refinement in partnership with analysts to help them externalize their intuitions about the data and thereby compute an appropriate, custom distance function.

Improving MDS Performance

We'd like to modify the MDS component to allow real-time user interaction with attribute weighting. In particular, we're interested in leveraging the research of Stephen Ingram and his colleagues, who reported speedups of 10 to 15 when using a GPU for their MDS algorithm.[10]

As the analysts in our experiments noted, the ability to perform multidimensional comparisons between simulation runs introduces the opportunity to examine agent-based simulation data in detail. However, owing to the lengthy computation time, analysts can't iteratively refine their comparisons by modifying the weight distribution across several variables and recomputing the distances between simulations. By refining and speeding up these calculations, we would enable analysts to better explore a range of hypotheses about the factors influencing sociopolitical interactions observed in their simulations.

Integrating MDSVis and SocialVis

Some analysts suggested combining the functionalities of MDSVis and SocialVis into one tool. Because of the memory management issues arising with such large datasets, such a tool would have to dynamically load data, potentially decreasing performance. However, the benefits of a combined tool that doesn't require analysts to switch contexts warrant further investigation into its development. This is especially true when these benefits are combined with the potential to dramatically increase performance by leveraging the GPU for MDS computation, which would offset some of the dynamic-loading bottleneck.

We're very pleased with our tools' effectiveness in analyzing and interpreting agent-based models for social and political science. We look forward to exploring their utility in analyzing the results of other agent-based model simulations from a variety of areas such as logistics and epidemiology.

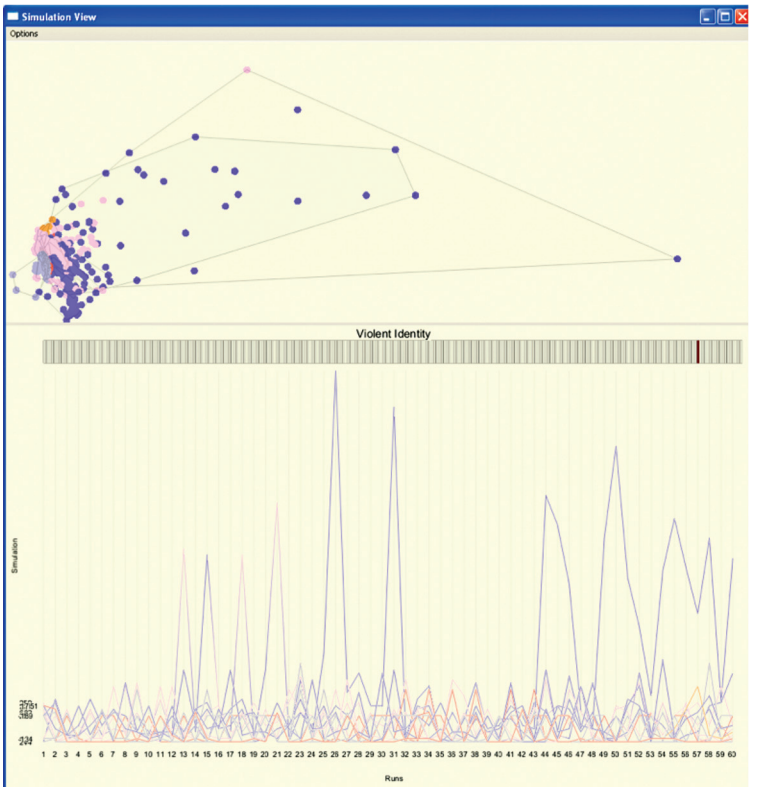


Figure 7. To analyze unlikely, yet high-impact outcomes, the analysts focused on outliers. Comparing two of these outliers to more characteristic runs, analysts used the temporal view to explore the attack attribute. The characteristic runs show little difference between the levels of attack, but the outlier runs show several spikes indicating high levels of attack.

Acknowledgments

The US National Science Foundation (NSF) supported some of this research under grant BCS-0904646. We also thank Ian Lustick, Miguel Garces, and Brandon Alcorn at Lustick Consulting. Data used in this research was provided with DARPA's support through the Advanced Technology Laboratories wing of Lockheed Martin in the Integrated Crisis Early Warning System project (Prime Contract FA8650-07-C-7749). This article has been approved for public release, distribution unlimited. Any opinions, findings, and conclusions or recommendations expressed in this article are solely the authors'.

References

1. R. Axelrod, *The Complexity of Cooperation: Agent-Based Models of Competition and Collaboration*, Princeton Univ. Press, 1997.
2. A. Srbljinovic et al., "An Agent-Based Model of Ethnic Mobilisation," *J. Artificial Societies and Social*

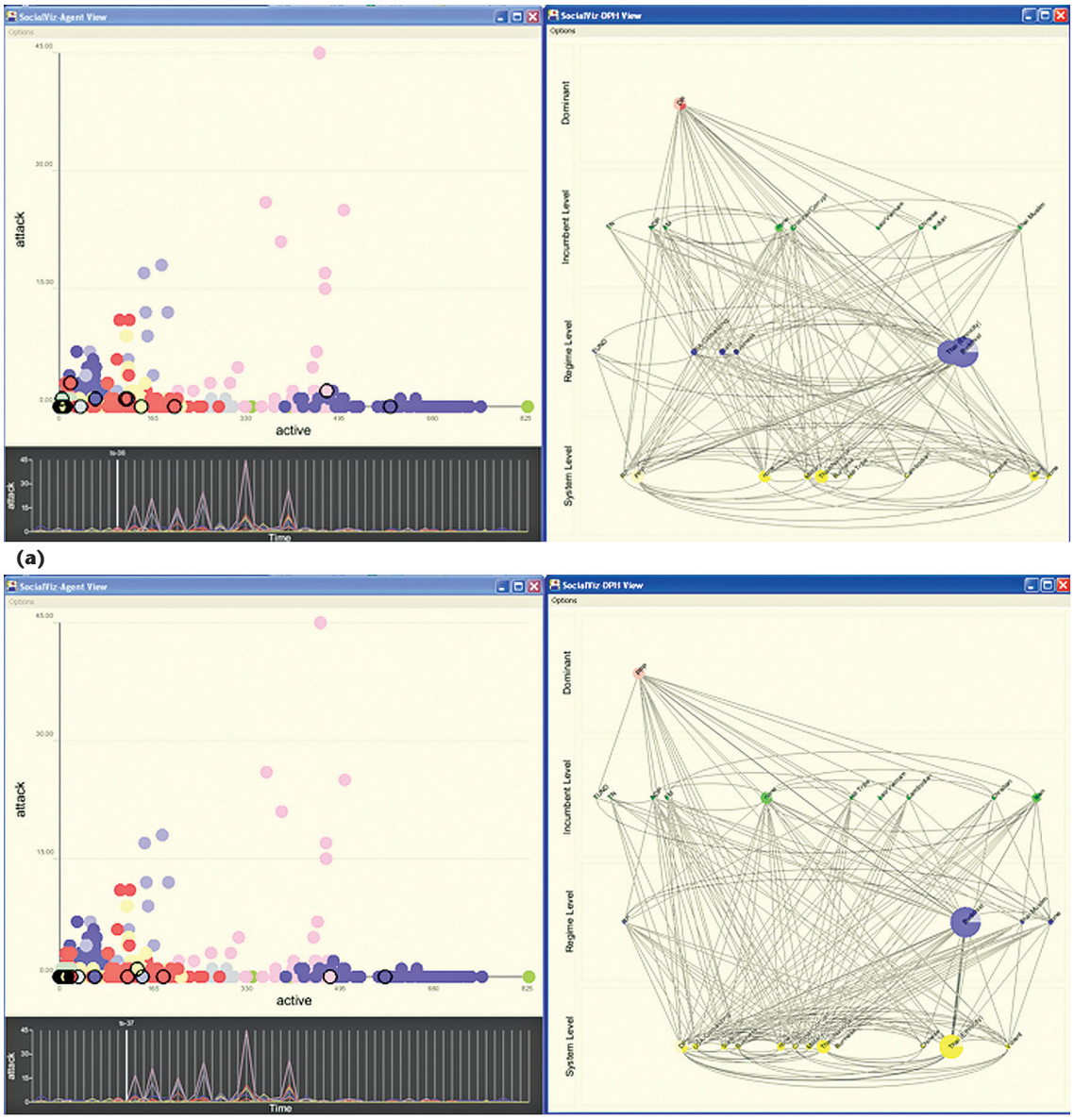


Figure 8. To understand spikes in attacks in two outlier runs, analysts examined the DPH view of the time steps immediately preceding the increase in attacks. In those time steps, the Thai Ethnic identity moved from the regime level (see Figure 8a) to the system level (see Figure 8b). Additionally, the Isan ethnic group moved from the system level (see Figure 8a) to the incumbent level (see Figure 8b). From this, the analysts concluded that, for this run, the high levels of attack probably resulted from the Thai Ethnic group's alienation whenever the Red Shirts aligned themselves with the minority Isan group.

Simulation, vol. 6, no. 1, 2003; http://jasss.soc.surrey.ac.uk/6/1/1.html.

3. R. Bhavnani and D. Backer, "Localized Ethnic Conflict and Genocide," *J. Conflict Resolution*, vol. 44, no. 3, 2000, p. 283.
4. R. Axtell et al., "Population Growth and Collapse in a Multiagent Model of the Kayenta Anasazi in Long House Valley," *Proc. Nat'l Academy of Sciences of the United States of America*, vol. 99, supplement 3, 2002, pp. 7275–7279.
5. I. Lustick et al., "From Theory to Simulation: The Dynamic Political Hierarchy in Country Virtualization Models," *Proc. Am. Political Science Assoc. 2010 Ann. Meeting*, Am. Political Science Assoc., 2010; http://ssrn.com/abstract=1642003.
6. I. Lustick, "PS-I: A User-Friendly Agent-Based Modeling Platform for Testing Theories of Political Identity and Political Stability," *J. Artificial Societies and Social Simulation*, vol. 5, no. 3, 2002; http://jasss.soc.surrey.ac.uk/5/3/7.html.
7. B. Alcorn, A. Hicken, and M. Garces, "VirThai: A PS-I Implemented Agent-Based Model of Thailand in 2010 as a Predictive and Analytic Tool," Lustick Consulting, 2011; http://lustickconsulting.com/?p=11.

8. D. Berndt and J. Clifford, "Using Dynamic Time Warping to Find Patterns in Time Series," *Proc. AAAI-94 Workshop Knowledge Discovery in Databases* (KDD 94), tech. report WS-94-03, AAAI Press, 1994, pp. 229–248.
9. J. Lin et al., "Experiencing SAX: A Novel Symbolic Representation of Time Series," *Data Mining and Knowledge Discovery*, vol. 15, no. 2, 2007, pp. 107–144.
10. S. Ingram, T. Munzner, and M. Olano, "Glimmer: Multilevel MDS on the GPU," *IEEE Trans. Visualization and Computer Graphics*, vol. 15, no. 2, 2008, pp. 249–261.

R. Jordan Crouser *is a doctoral candidate in Tufts University's Computer Science Department. His research interests include human computation, human-computer teams, visual analytics, and human-computer interaction. Crouser has an MSc in computer science from Tufts University. Contact him at rcrous01@cs.tufts.edu.*

Daniel E. Kee *is a senior software engineer at iRobot. His research interests include visual analytics and tangible user interfaces. Kee has an MSc in computer science from Tufts University. Contact him at dan.kee@tufts.edu.*

Dong Hyun Jeong *is an assistant professor in the University of the District of Columbia's Department of Computer Science and Information Technology. His research interests include information visualization, visual analytics, human-computer interaction, and information security. Jeong has a PhD in computer science from the University of North Carolina at Charlotte. Contact him at djeong@udc.edu.*

Remco Chang *is an assistant professor in Tufts University's Computer Science Department. His research includes visual analytics, information visualization, urban modeling, and computer graphics. Chang has a PhD in computer science from the University of North Carolina at Charlotte. Contact him at remco@cs.tufts.edu.*