Presented at the 14th Annual Conference for the
North American Association for Computational
Social and Organizational Sciences 2006

A Framework for Modeling Social Power Structures

Glenn Taylor, Robert Bechtel, Geoffrey Morgan
Soar Technology, Inc.
3600 Green Court Suite 600, Ann Arbor, MI 48105
{glenn,bob.bechtel,gmorgan}@soartech.com

Ed Waltz
BAE Systems
3811 N. Fairfax Drive Suite 500, Arlington, VA 22203
ed.waltz@baesystems.com

Abstract

This paper describes an agent-based framework for building models of social power structures. Two goals of the framework are generality across a wide range of power structures and the ability for domain expert end-users to build, run, and analyze models of power structures in their particular domain. We describe the framework, its motivation and design, and discuss our experiences in building models in this framework. Finally, we discuss some lessons learned in building a framework meant for end-user modeling and simulation of social networks.

Contact:
Glenn Taylor
Soar Technology, Inc.
3600 Green Court, Suite 600
Ann Arbor, MI 48105

Tel: 1-734-327-8000x205
Fax: 1-734-913-8537
Email: glenn@soartech.com

Key Words: power structures, agent-based modeling, social networks, end-user modeling

This work was performed under DARPA contract W15P7T-05-C-P032. This paper has been approved for public release, distribution unlimited.

A Framework for Modeling Social Power Structures

Glenn Taylor, Robert Bechtel, Geoffrey Morgan, Ed Waltz

This paper describes an agent-based framework for building models of social power structures. Two goals of the framework are generality across a wide range of power structures and the ability for domain expert end-users to build, run, and analyze models of power structures in their particular domain. We describe the framework, its motivation and design, and discuss our experiences in building models in this framework. Finally, we discuss some lessons learned in building a framework meant for end-user modeling and simulation of social networks.

Introduction

Scenario planners are increasingly concerned with understanding the social networks that make up a particular real or hypothetical situation, including who the players are, be they individuals or influential groups, the interconnections and dependencies in the network, and how the network's character changes over time. In this paper, we use the term *power structures* to describe these kinds of networks. What is of special interest is understanding how influences (actions) on those networks will have impact, both immediate and cascading in ways that are not obvious, across a range of domains of activity, such as in social, economic, and political realms. For example, what happens to a group if its leader is arrested? Or, what is the impact on the economy if a segment of the population is flooded with money?

This paper describes a modeling framework and a graphical toolkit for building models to help answer these kinds of questions. This system, the Power Structure Toolkit (PSTK), is meant for domain experts, rather than experts in simulation, which presents certain challenges when building a system. In this paper, we describe the PSTK framework, its motivations and design. We also discuss our experiences with building models in this framework, and draw some conclusions from this effort.

Power Structure Toolkit (PSTK)

Project Goals and Approach

There are two major goals that have driven the design of the Power Structure Toolkit. First, the PSTK must be capable of representing a range of power structure models, covering anything from national political parties, social or religious leaders, to organizations. Second, a domain expert, rather than a computer scientist, must be able to build power structure models and analyze their dynamics.

End users in this application might typically be intelligence analysts or military planners, who must be able to develop, execute, and analyze their own models without writing source code, as is typically the case in model building. This could be addressed in a number of ways, perhaps on one end of the spectrum implementing something like a visual programming tool where the normal "programming" methods are cast in a way more appealing to non-programmers. At the other end of the spectrum is to fix some constructs and processes, but allow (some of) those processes to be parameterized through a user interface. This latter is the method chosen here.

First, to enable building a large class of models, we defined an abstraction of actor elements and interactions that could reasonably represent a range of power structure models, and which fit with some accepted frameworks of organizational behavior, but which were simple enough to understand. Second, we developed end-user tools that supported use cases such as building, running, and analyzing models defined in this framework. The first enables the second: a single, fixed level of abstraction that applies to individuals and organizations alike is more easily represented in a simple user interface with fixed data structures. Of course, abstraction is a double-edged sword. On the one hand, it allows for generalization and simplification, but on the other it results in loss of fidelity when trying to model specific instances. The remainder of this section describes the PSTK framework in detail.

PSTK Simulation Framework

PSTK represents a hybrid of stock-and-flow network simulation and agent-based decision making, where agents have control over outflows from stocks. The architecture extends a conventional stock-flow system modeling approach by adding the following characteristics:

- Goal-directed behavior – actors behave in ways that attempt to accomplish their goals
- Decision theoretic – an agent's choice of action is optimal based on its beliefs about how its environment works
- Context-dependent – the actor's behavior (selection of goals and actions) is based on the situation at any given time

For this application, we are interested in the decision-making of specific agents, and the second- and third-order effects of an agent's decisions, rather than swarm-like behaviors of groups. In these models, an agent can represent an individual, a small group, or a population – with the assumption that there is a certain amount of homogeneity and consistency in terms of goals and beliefs within these representatives such that we can model them usefully within an agent that itself has the same consistency and homogeneity. This drove us to look at a certain class of agent-based simulation, one consistent with the beliefs-desires-intents (BDI) paradigm [Bratman 1987].

The PSTK provides an *architecture* for modeling social power structures. As an architecture, it provides a set of fixed constructs – structures and processes – to the models built within the architecture. These structures and processes provide the lexicon with which to build models, and provide the means by which the model executes. These fixed constructs are as follows:

Actors. Represented as software agents, Actors are decision-making entities, and can be used to represent organizations, population groups, or individuals, such as group leaders.

Process Blocks. Process Blocks are entities that do not make decisions. They share most of the properties and operations of actors, but follow a fixed behavior pattern.

Domains. Each actor and process has a set of domains and subdomains. *Domain* is the PSTK term for a type of stock. Domains may have *subdomains*, which are either *capital stocks* or *abilities*. Domains and capital stock subdomains have values ranging upward from zero, while ability subdomains have values between 0 and 1. In the PSTK, the basic domains are *political, military, economic, and social.* Each of these domains have different categories of subdomains, such as *leadership acumen (political),* or *armed mobilization (military).*

Effective Capital. The *effective capital* of a domain is calculated by adding together its capital stock subdomains and multiplying that result by the average value of its ability subdomains. The effective capital is the actual amount of capital an actor can spend to achieve its goals.

Lines of Influence. Entities (actors and processes) are linked by *lines of influence* or LOIs. Each LOI is directed, with a source entity and a sink entity, and is associated with one top level domain. If an LOI exists between two entities, then capital in the associated domain can be passed from the source entity to the sink entity. A transfer can be positive or negative. A positive transfer results in a decrement of the transferred amount from the capital stock of the source entity, and an increment of the transferred amount to the sink entity. A negative transfer results in a decrement of the transferred amount from both the source and sink entities. This transfer of capital over LOIs is the only means by which agents interact in PSTK models. In the current PSTK implementation, the entity-LOI network structure of the model cannot be changed by the agents while running. Lines of Influence can be left unused, but no new lines can be created during a run.

Goals. Each actor has a set of *goals*. A goal is simply a statement of the desired state relative to the subdomain of some actor, and may be either self-referential or refer to a subdomain of another actor. The desired state may be expressed relative to an explicit numeric value or to the current value of the subdomain. So, for example, actor A could have goals such as, "increase A's popular support" or "make A's wealth greater than B's wealth."

Contexts. Goals are active in certain *contexts*. A context is much like a goal in that it describes a state of the world, e.g., A's wealth is greater than B's wealth, but acts as a precondition for a goal to be in use. This allows for certain adaptiveness to the environment –goals are active in particular contexts, and inactive in others. This shifting of goals with situations can be used to model different types of behavior.

Belief Network. Each actor has a belief network about the structure of the social network in which it exists, which allows it to make decisions to meet its goals. Beliefs are used to determine a set of actions that might positively impact the selected goal, and the best action is selected based on utility (e.g., most likely to affect desired change) and preferences for that action. An actor's belief network is its own model of the entity-LOI network, with the LOI tagged as positive or negative (termed "valence") to indicate the actor's belief regarding the net flow over that LOI. By default, each actor gets a belief network that matches the top-level model network, but that default can be varied by the model builder. For example, an actor can be unaware of other actors (they will be missing from the belief network), can believe that LOI exist that are not in the model-level network, or can fail to believe in LOI that

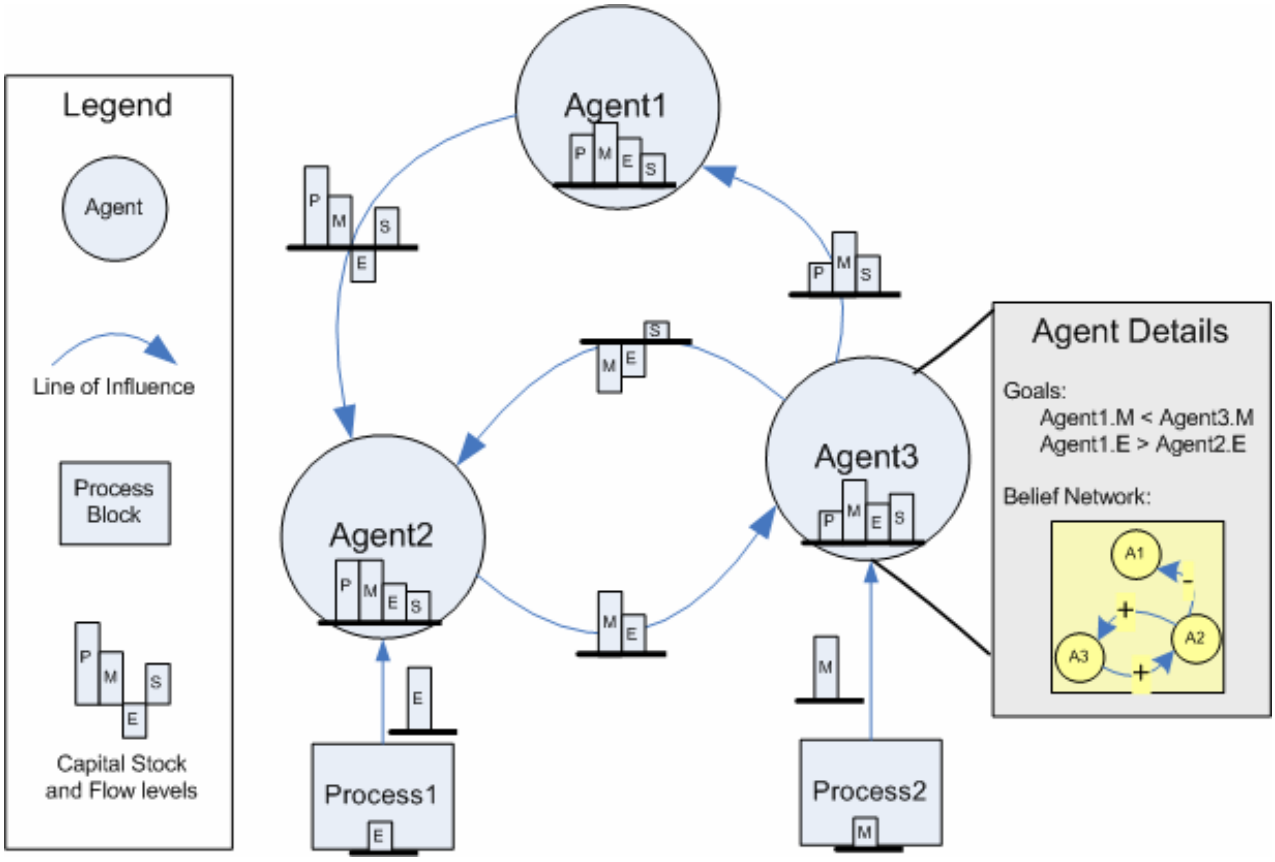

Figure 1: The PTSK Modeling Components

exist in the model-level network, or can have a different valence on LOI. Note that there is a ground truth belief model (called the *World Model)* that is used to compute the universal next state each agent perceives.

Turns. The system operates in a turn-based manner. A *turn* consists of a full cycle of all agents transferring capital to other agents over their available lines of influence. All transfer decisions are calculated, then all transfers take place instantaneously, resulting in updated capital stock values for all entities on the next turn. On each turn, the actors decide how much capital to transfer along their outgoing LOI. To do so, they reason over three other model components: Goals, Contexts, and Belief Networks.

Events. Within a simulation, the user can schedule changes to take place at specific turns. These changes represent effects of external actions to the model, such as changing the value of a stock, ability, goal, or LOI suddenly or over time. This allows the user to play 'what if' games to see the cascading effects of actions on a model.

Figure 1 illustrates these major concepts. Actors have particular capital stocks in the four domains (PMES). Actors are connected by various Lines of Influence, which represent potential channels to transfer agent-decided amounts of capital in the four domains. Process Blocks serve as "pumps" that push capital of any domain into the system. There are a few additional concepts included in the architecture.

Model Execution

At the start of each turn, the world model state is computed, based on the state of the previous turn, any agent actions that have been registered, and any events that are scheduled. The results of this computation (the new world state state) are passed to the agents, where each agent can see the capital stock domain values of itself and other agents (lines of influence and domain abilities about other agents are hidden). From this, each actor begins its work,

starting with its highest priority goal whose context holds. It determines if the goal is currently satisfied. If so, it goes to the next highest priority goal. If not, the unmet goal becomes an issue to deal with, and the agent consults its belief network to determine if there is an LOI pathway on the relevant subdomain between itself and the subject of the goal. If so, it commits a portion of its available effective capital to the outbound LOI that is the first link on that pathway. (There is reasoning over multiple pathways, with shorter paths preferred, and lower intermediate fan-out preferred.) Currently, the agent deals with only a single issue per turn. At the end of its processing, the agent signals that its turn is done. When all agents have completed, the process begins again with the computation of a new world state.

User interface

The current native PSTK user interface is a combination of a custom GUI for building agent networks, goals and beliefs, and MS Excel for viewing model output. (Model results are dribbled out to a .CSV file.) While this is a rudimentary start, we already have had intelligence analysts modifying and analyzing models with some success, without writing code. From previous experience, a great deal of time and effort can be put to an end-user GUI for building models, and this needs to be balanced against building a complete framework that is usable from the bottom up. Figure 2 illustrates a representative model within the GUI. We have also connected the PSTK to an external data viewing tool [Wright et al. 2005] that allows for easy navigation of time-series data and correlation of those data with other kinds of information.

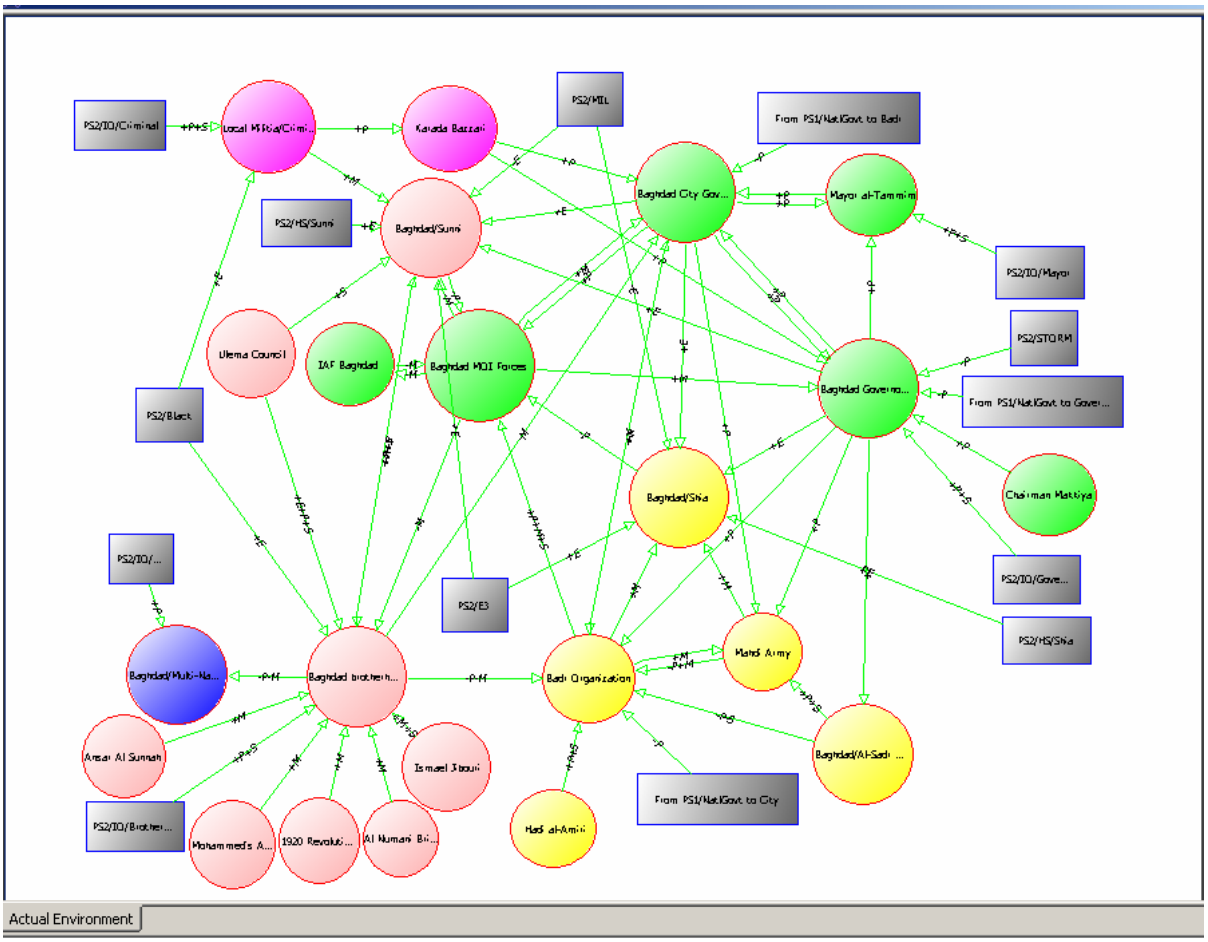

Figure 2: An Example Model in the PSTK GUI

Discussion, Conclusions, Future Work

One major design tradeoff is between simplicity of the user interface for end-user model development, and an expressive modeling framework capable of interesting single and multi-agent behavior. Especially in the kinds of models we are interested in, where there might be known interaction protocols or behaviors, more complex agents are required, yet such agents are difficult to specify without writing code. For now, we have erred on the side of simplicity in usability and, therefore simplicity in agent capabilities: all agents have the same kinds of interactions, regardless the structure or implied relationships between agents. In some cases, we have had to place assumptions in the architecture about agent decision-making, making them an immutable part of the architecture until we find reasonable means to allow a non-developer to affect those behaviors. One possibility along these lines is to have well-defined behavior modules that an end-user can combine without needing to build those modules manually. The use of modules, and their assumptions, would have to be very clear to the user, and the rules for combining behavior modules would have to be very well-defined. End-user behavior composition of this sort can be a thorny issue, with several research groups actively pursuing it.

We have designed the PSTK to fit within a larger, multi-paradigm simulation environment through the use of a simulation backplane. When interacting with other simulations, the Process Block entities can act as portals for passing information back and forth between models. For example, while the PSTK might model the power structure among decision-makers in an environment, other models might represent the health infrastructure, or the electrical power grid, whose effectiveness may have an impact on the power structure (e.g., lack of electricity acts as a dampening effect on the ability of actors to employ political capital stocks).

One acute challenge in modeling and simulation is model validation. Because we are interested in realistic models of existing networks, we have a high standard to meet in terms of building models that represent actual human networks. Compounding this challenge, the models built in this framework are highly abstract, and if real data are available, they are not of the same character as the model data. Furthermore, often in analytical domains, data are subjective, interpreted, filtered, etc., rather than being purely qualitative. Converting these available data to data in model terms is itself fraught with bias. One area where data are often unavailable is in models of decision making. Here we rely almost entirely on the expertise of analysts to advise the development of models, and in the process of face validation of the results. Validation in general remains an open issue in modeling and simulation.

Another question of validity is simply, what do the models built in PSTK represent? For example, what does a political capital stock level of 200 mean? Do the levels really matter, or is the meaning in the value relative to another value (e.g., Actor A's capital stock versus Actor B's, or Actor A's change over time)? This question is particularly important when trying to model specific real instances, where terms used in the model must have explainable analogs in the real situation. In many ways, the meaning of the model is imparted by the model builder – he or she must have a sense of the meaning of "political capital = 200" within the system, and those design decisions at the modeling level must be available and explainable to outward observers. The models we have built to date include such design rationale as part of the model description. As with all modeling efforts, the kinds of questions one wishes to ask of the model needs to drive the design of the model. In some models, relative valuations between actors may be more important than the actual levels of stocks.

Future Work

We are just now beginning to experiment with this framework in a meaningful way. We have built a number of models, and have plugged them into a simulation backplane to interact with other non-PSTK models. The framework itself allows means to experiment at the *model* level – building multiple models each representing a hypothesis, and playing with those hypotheses by running, setting different initial conditions, different agent networks, different agent goals and beliefs, and injecting different events. We have yet to build tools in PSTK that directly support this process of hypothesis testing, though earlier examples these tools is discussed in [Taylor et al. 2004].

There are a number of dimensions within the architecture that merit further investigation, such as the agent decision-making process, the stock-and-flow model that assumes accumulation of stock over time (versus non-accumulating), etc. A rich area of investigation is online model adaptation, as is typical of other complex adaptive system frameworks. In the PSTK, goal contexts represent the only form of intrinsic model variation while running (different from events to model extrinsic changes). Future work includes incorporating a means for belief learning, goal adoption, coalition forming, etc. Also, adding additional capabilities to the agents, such as the ability to discover or adapt the network structure will be interesting extensions.

Foundations and Related Work

The PSTK modeling framework does not embody a single theory, per se. Instead, it is inspired by work in social science. Graham Allison's Rational Actor Model [Allison and Zelikow 1999] forms the basis for agent decision-making. Michael Mann's [Mann 1986] theory of Social Power influenced the choice of the PSTK domains of influence – Mann defines four primary sources of power: *ideological, economic, military,* and *political (IEMP).* The agent system, built using the Soar cognitive architecture [Laird et al. 1987], is designed along the lines of BDI architectures [Bratman 1987].

In terms of extant systems, this work is a direct descendant of AGILE [Taylor et al. 2004], a framework for building agent-based models of nation-state conflict. Many of the lessons learned from AGILE were applied here, especially in the area of end-user model building. Like AGILE, the basis for this work is Soar, which provides a parsimonious architecture for building goal-directed agents capable of both deliberative and reactive behaviors. PSTK, from an agent capabilities perspective, is a pared-down version of AGILE, though is more flexible in the power structures that can be modeled. AGILE and PSTK both have roots in earlier work in applying a cognitive architecture (Soar) to social modeling [Carley and Prietula 1993].

References

[Allison and Zelikow 1999] Allison, G. and Zelikow, P. (1999). Essence of Decision: Explaining the Cuban Missile Crisis, Longman.

[Bratman 1987] Bratman, M. (1987). Intention, Plans, and Practical Reason. Cambridge, MA, Harvard University Press.

[Carley and Prietula 1993] Carley, K. M. and Prietula, M. J. (1993). Plural-Soar: Towards the Development of a Cognitively Motivated Theory of Organizations. Coordination Theory and Collaboration Technology Workshop, Washington, D.C.

[Laird et al. 1987] Laird, J. E., Newell, A. and Rosenbloom, P. S. (1987). "Soar: An architecture for general intelligence." Artificial Intelligence 33(3): 1-64.

[Mann 1986] Mann, M. (1986). The Sources of Social Power, Cambridge University Press.

[Taylor et al. 2004] Taylor, G., Frederiksen, R., Vane, R. R. I. and Waltz, E. (2004). Agent-based Simulation of Geo-Political Conflict. Ann Arbor, MI, Soar Technology, Inc.

[Wright et al. 2005] Wright, W., Schroh, D., Proulx, P., Skaburskis, A. and Cort, B. (2005). Advances in nSpace -- The Sandbox for Analysis. Intelligence Analysis, McLean, VA, Mitre Corporation.