

Enabling Autonomic Computing by Building Cognitive Maps of Complex Computing Systems

Prof. Benjamin J. Kuipers
Department of Computer Sciences
The University of Texas at Austin

IBM Technical Contact - Dr. C. J. Paul
Tivoli Technical Strategy, IBM Software Group
7 December 2001

Introduction

IBM's goal of autonomic computing requires the creation and use of intelligence technologies to manage large scale computing environments. Systems built with this approach need to adapt to complex changing environments where the control laws are not known a priori. The system will have to learn its environment and the context surrounding its activity and act accordingly. To build such systems, we propose to adapt our research and expertise in building computational models for robot exploration, and apply that to the domain of managing complex computer systems, using an Apache web server and Tivoli monitoring software as an experimental testbed.

An expert in a complex system is said to have a good "cognitive map" of that system. This metaphor draws on the human ability to explore an environment and build a cognitive map that supports effective navigation through the space.

In previous work, we have developed a successful and influential computational model of the human cognitive map, and applied it to the problem of robot exploration, mapping and navigation. We propose to apply these techniques to build "maps" of more abstract spaces such as the output of complex scientific models and the parameter spaces of complex computing systems.

The human cognitive map includes many different representations for different types of spatial relations, generally organized into four layers:

- Control: continuous control laws for hill-climbing to locally distinctive states or following trajectories from one distinctive state to the neighborhood of another.
- Causal: an abstraction of continuous change to discrete actions linking widely separated distinctive states.
- Topological: a qualitative description of the space in terms of places linked by paths, with both organized into a hierarchy of regions.
- Metrical: local and global models of quantitative relations among states in the state space.

The key benefit of the multiple layers of description is the ability to represent partial knowledge, allowing incremental inference and successful behavior even when knowledge is incomplete [Kuipers, AIJ, 2000]. We have also been developing methods for learning such representations without a priori knowledge of the nature of the robot's sensors, effectors, or environment [Pierce & Kuipers, AIJ, 1997].

A complex computer system produces a mass of data that is near-impossible for the unaided human to comprehend. If we can apply this concept of "cognitive map" to such a body of data, the resulting model will give informed guidance to human engineers monitoring or studying the system, and to automated observers that need to track where in its state-space the system is or how to get it somewhere else.

In effect, we are proposing to develop a kind of "knowledge-rich data-mining" that attempts to identify an instance of a complex model (the cognitive map) in a large mass of behavioral data. The fact that humans find spatial metaphors so useful suggests that this is an achievable goal.

Technical Progress and Goals

In an important line of related research, Joseph Hellerstein and his colleagues at IBM Research in Hawthorne NY have demonstrated that traditional design methods from control theory can be applied to design high-quality SISO and MIMO controllers for the Apache web server. These design methods assume that the system being controlled is linear, which is manifestly untrue for most computer systems, yet their controllers provide an impressive quality of control.

Most nonlinear systems can be locally approximated by linear models, so controllers designed under the assumption of linearity can work quite well in local regions, but will fail as the system state departs from that neighborhood. In our approach to the cognitive map, we use local models to define control laws that converge to "distinctive states", and allow the global non-linear behavior to be described by the causal and topological graphs linking the local models.

Within our framework, we hypothesize that Hellerstein's controllers can be regarded as the control laws that keep the system near a "distinctive state". Our first technical goal will be to determine what changes (if any) to our framework are required to unify the technical concept of distinctive state in our framework with the setpoints of the controllers designed in Hellerstein's work. The second technical problem is to determine how different distinctive states and their neighborhoods are linked: (a) by the natural dynamics of the system, or (b) by the decision to apply a particular control law, or (c) by an external perturbation that is too large, or too unexpected, to be kept near the current distinctive state by the local control law.

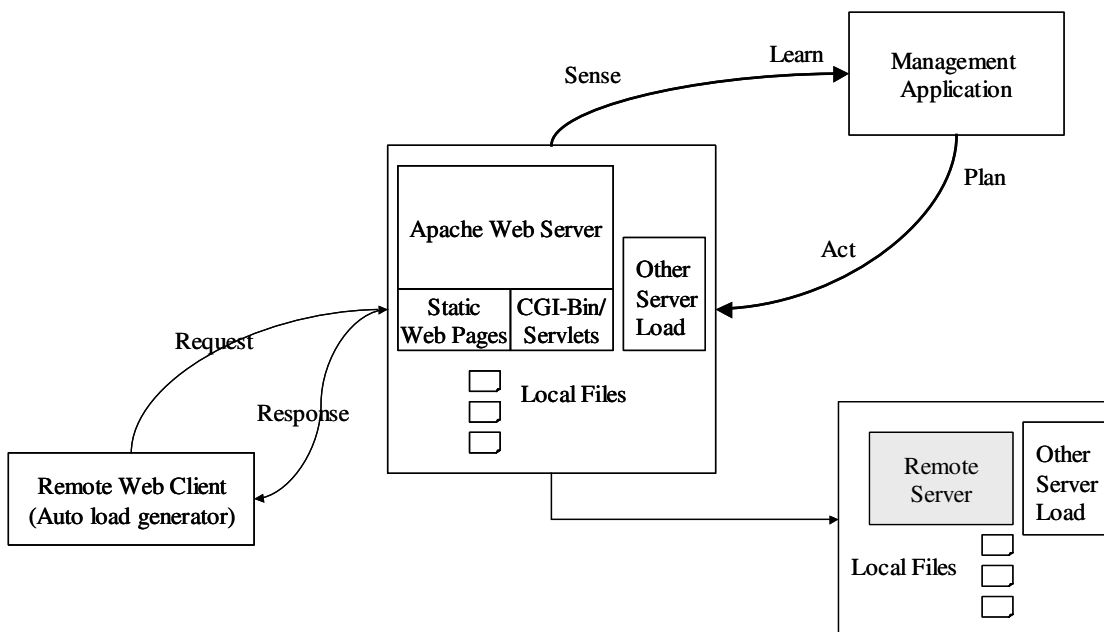
In our current efforts on this second problem, we are exploring an abstract model system with two distinct behavioral modes and transient transitions between them. The less desired mode is a stable attractor in the phase space of the system. Our goal is first, to identify the properties of the natural behavior of the system in that mode, and second, to design a controller to destabilize that mode and force a transition to the other mode.

The more desired mode is an unstable fixed-point (a saddle) in the phase space of the system. Trajectories coming arbitrarily close to this fixed-point will still follow the transition back to the other mode. One subgoal is to identify the properties of this behavioral mode. Note that, although the system's designers presumably intended the more desirable mode, even its existence is not entirely obvious from behavioral data. Once a local dynamic model for the desirable mode has been identified, the other subgoal is to design a controller to keep the system near it, in spite of perturbations.

The methods we need for this are methods for system identification, especially in the face of incomplete, qualitative knowledge of the properties of the system or of the observed data. Such methods have been developed in Bradley & Stolle's PRET, in Kraan, Richards & Kuipers' MISQ [AAAI-92], and in Kay, Rinner & Kuipers' SQUID [AIJ, 2000]. We will draw on these methods and adapt them as needed for our systems.

Our scientific goals for exploration of this model system are (1) to create a general framework combining cognitive maps and qualitative dynamical systems for describing such systems and their controllers, (2) to determine when and whether the "cognitive map" metaphor translates to a technical description --- that is, whether complex computer systems actually have "distinctive states" and transitions between them --- and (3) to develop a set of general-purpose tools that will be useful for identifying higher-level features from low-level observational data, that can be used to build useful control laws, which then define distinctive states that can be linked together into cognitive maps.

The major day-to-day effort will go into the development of the tool-set for data analysis, which will be general-purpose, intended for both complex computer systems and mobile robots. We already have several mobile robots in our laboratory. We will collaborate with researchers from Tivoli to set up a small test network running the Apache web



server and several load generators in our laboratory, to allow us to test the validity of our approach on a realistic but tractable model of a complex computer system. We intend to use load generators on a remote web client to generate traffic load into the web server. The web server satisfies these requests with either static web pages, or dynamic web pages built with CGI scripts or servlets. The dynamic web pages aggregate content from local files and remote files provided by a remote server. Various load generators are sprinkled throughout the environment to simulate other programs that may be running on those servers.

The scientific principles will be demonstrated within the context of a simple management application that is collecting data from the environment, analyzing it, learning from it, integrating new knowledge with pre-existing knowledge, and then using its combined knowledge to configure the properties of the web server to maintain desired operational characteristics.

Thus, using this testbed, we expect to demonstrate how AI technologies can be applied to support the goals of autonomic computing in the domain of systems management.