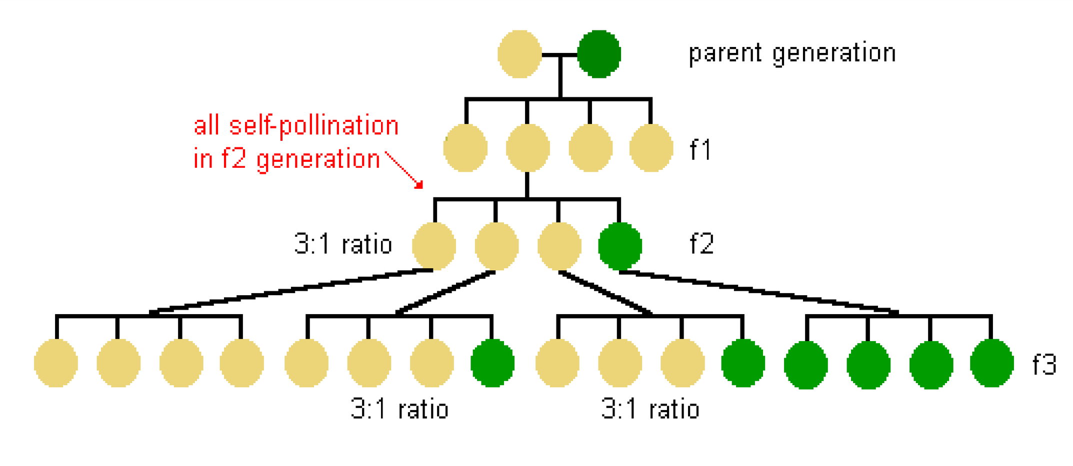
Neural Models of Mind: Mapping Reflective Computation

Bo Morgan
bo@mit.edu

Run-time reflective computation is a field of computer science that allows processes to be watched by other processes as they are running. Understanding natural processes as computational models has proven to be a useful way of seeing and simulating the world around us. If the computational model is simple enough, such as Mendel's binary model of genetic inheritance, it can be simulated within a human mind, such as Mendel's mind. However, when the computational processes become complex, such as models of world economies or human minds, they become impossible for humans to mentally simulate without computers. Measuring features of the natural process of cognition as evidenced by the human brain have become more numerous recently; these include: fMRI, EEG, MEG, PET, fNIRS, and others. In addition, secondary external natural features include: EKG, EMG, GSR, and others.

Example: Simple Computational Model



Mendel's Model of Genetics

However, useful computational features have been more illusive. Examples of the most basic computational features include: (1) **memory creation**, (2) **memory read**, and (3) **memory write**. Tracing all causal relationships between these basic features **allows tracing the context of all other programmer-defined semantic abstractions**. All of these computational features create an intricate trace network of dependencies, automatically traced and shared by many parallel threads of execution. We are experimenting with a programming language that allows this to occur modularly to dynamically chosen parts of large complex processes. The resulting dependency trace networks can be processed by critically reflective threads. Discovering useful reflective threads for cognitive models of learning in complex environments is one of our goals.

Reflective critics help a planner to learn to plan through run-time experience.

Cooperative Resource Selector Learners

As goals are often pursued and accomplished together these groups can be recognized and remembered for future planning deliberative simulations.

History Writers

Patterns in traces can be recognized and compiled into simpler representations for other critics to process. For example, all causal dependencies relevant to accomplish a specific goal can be compiled for quick retrieval later.

Conflicting Resource Allocation Learners

As conflicts are traced between groups of goals, these goals can be learned to be within mutually exclusive allocation sets. These MUTEXs can be learned critically through run-time trace feedback.

Resource Conflict Blame Arbiters

If there is a problem attempting to allocate a lower-level resource for two independent threads of execution, a critic may attempt to discover what two subgoals are to blame for this unanticipated interaction.

Problem Distribution Balance Learner

Many resources are limited, forcing the serial execution of some goals. We can recognize that some combinations of goals are better than others for either optimal resource distribution or a minimal time until goal completion.

Pointless Process Recognizer

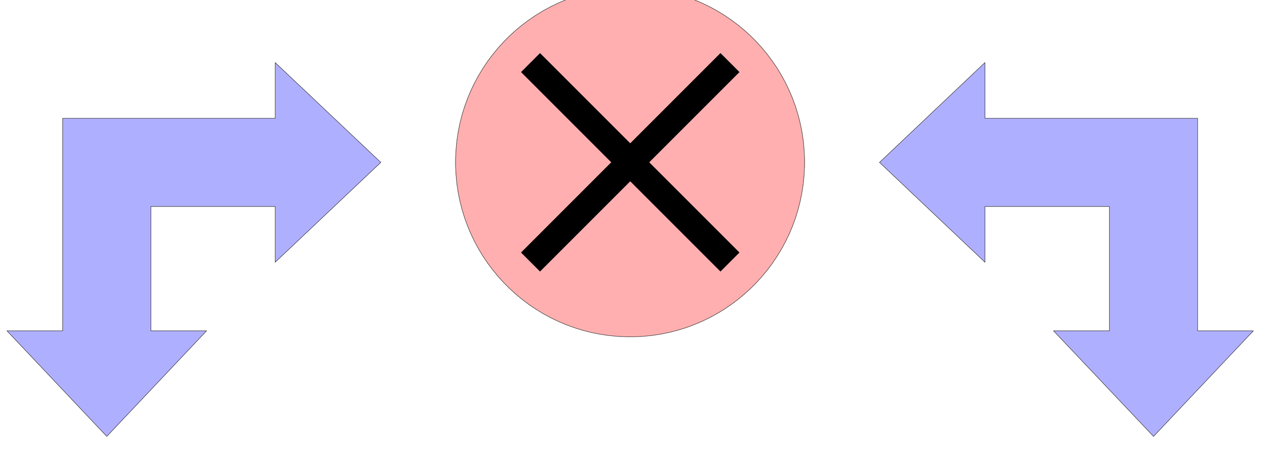
If there is a process that executes and ultimately has no effect toward accomplishing the goals of the system, note that these processes did not need to be executed.

The ability to reflect on causal dependency traces of a computational process allows two things:

1. Begin **mapping** natural features to and from computational features.
2. Begin **designing** novel reflective computational learning models of cognition.

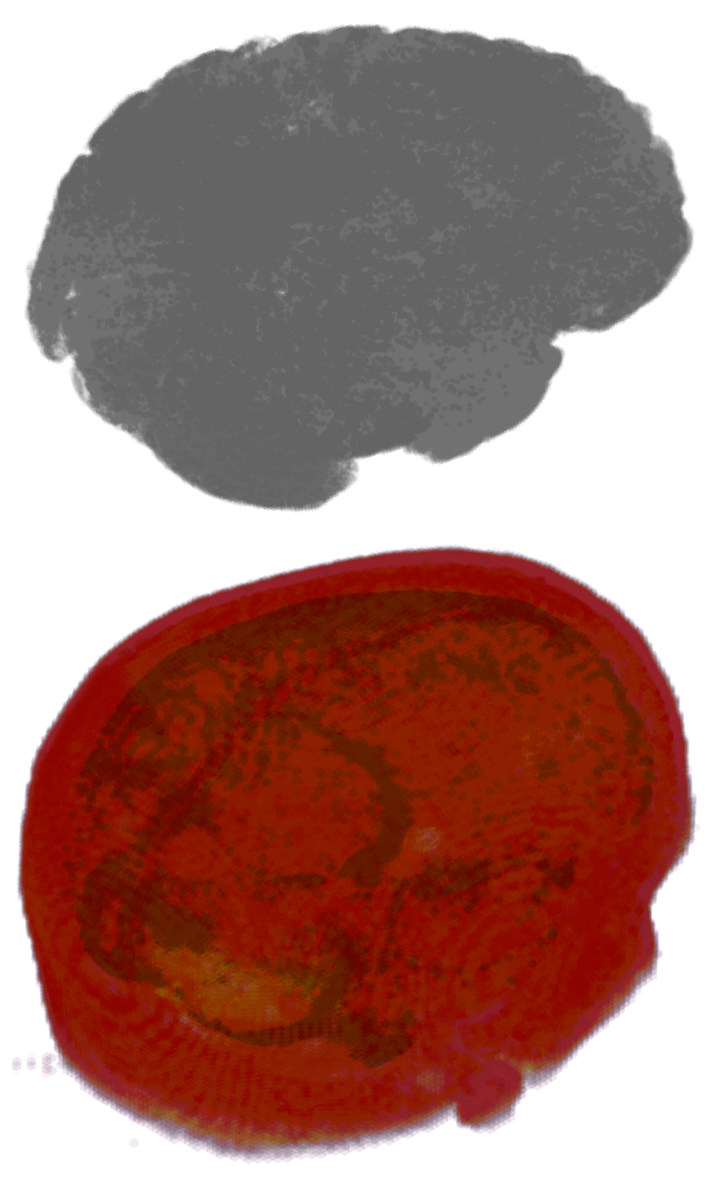
Correlation of Natural with Computational Features

Many features of natural processes can be measured, but it is currently difficult to correlate these features with complex models.



Very few features of computational processes are currently measured. We are writing an experimental programming language for measuring these features.

Example: Nontrivial Natural System



Natural Features:

fMRI,
EEG,
MEG,
PET,
fNIRS,
EKG,
EMG,
GSR,
etc.

Example: Nontrivial Cognitive Architecture

Preferentially Ordered Declarative Goal Structures

A dynamic goal structure distributed throughout a network of interconnected parallel computational problem solving resources.

Imagined Plans: Cooperative Subgoal Collections

Collections of sufficient subgoal conditions for comprising modular components of the overall distributed declarative structure.

Memoized Mental Resource Simulators

Each actor's execution can be memoized dependent on goal structure context, which can be used for simulation without external effects.

Traceable Compiled Mental Resource Actors

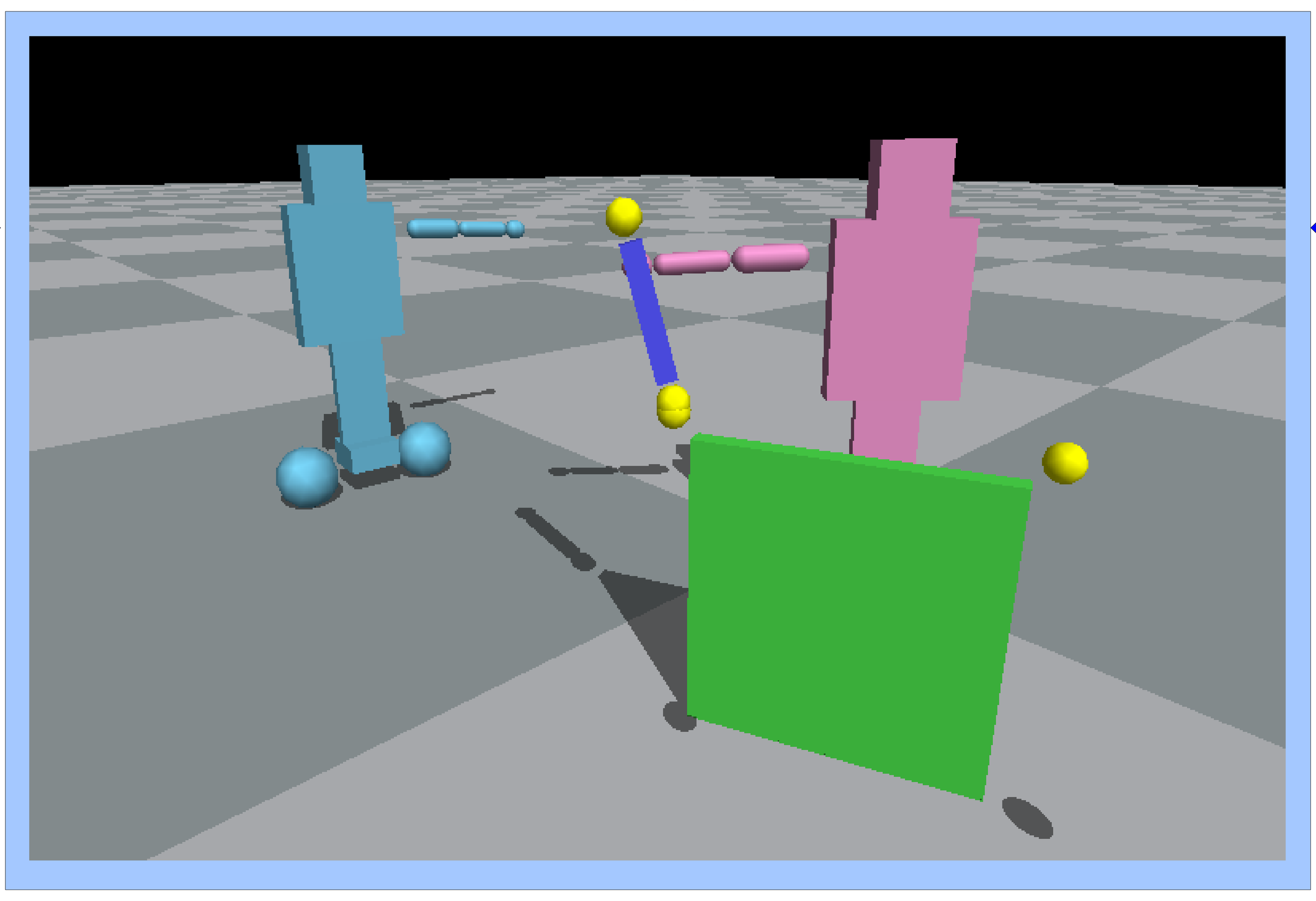
Each sequential effect of these traceable actors is recorded, such that if any error occurs debugging processes can know what is responsible.

Trusted Compiled Mental Resource Actors

Repeated successful execution of traceable actors results in the creation of trusted actors—optimized and compiled for fast execution.

PERCEIVE and ACT

Example: Nontrivial Cognitive Environment



PERCEIVE and ACT

Natural biological humans and their very complex brains can act within similar behavioral experiments as human-designed computational models of intelligence (A.I. models).

Human-designed computational models of intelligence (A.I. models) can interact with simulated problem-solving environments of nontrivial complexity.