RC Research Fellowship Proposal: OpenBrain

William Guss
26793499
wguss@berkeley.edu

January 3, 2016

1 Background

2 Goals

There are essentially four goals of the OpenBrain project.

- Build a massiveley parallel Beowulf cluster of parallela computers controlled using MPI on ArchLinux.
- Create an always online, turing complete modification to the recursive neural network algorithm whose fitness is determined by the Universal Intelligence Measure described in (Legg and Veness, 2011). The algorithm must have the following constraints:
  - In the spirit of John Conway's turing complete Game of Life, the individual neural nodes must follow arbitrarily simple rules in a decentralized fashion.
  - *Training* is unsupervised and occurs over the lifetime of an *instance* of the open brain, such that the aforementioned governing rules are modified with respect to the fitness of the instance.
- Implement each neural node as an Erlang process distributed across the Beowulf cluster asynchronously.
- Provide always on input/output to the OpenBrain cluster in similar fashion to that done in Google DeepMind's Deep Reinforcement Learning.

3 Plans

The OpenBrain project will commense in two phases. First the algorithm in question must be exactly theorized and mathematical guarentees about its capabilities must be given. Then given that motivation, the project must be implemented in hardware and software.

For the theory behind OpenBrain, we apply the nuronal model proposed in (McCulloch and Pitts, 1949) and assume that each neuron functions according Conwaynian turing complete rules; that is in particular, we apply the synaptogenesis model suggested in (Thomas et al, 2015) with parameters dictated by a universal intelligence indicator function.

For any neuron π in the OpenBrain algorithm, a list of axonally connected posterior neurons P_π is stored along with a list of proximal neighboring neurons, N_π. The neuron π is efficiently implimented as asyncrhonous thread with a message queue such that neurons connected to its anterior can activate and provide a voltage on π such that π itself will activate. Furthermore, each neuron has the capacity to signal to a particular proximal neuron in order to form a new dendritic or axonal synapse.

Under this lightweight model, we will experiment freely with different schemes of synaptogenesis and learning. Upon finding the most appropriate model, guarentees will be made on its computational capacity of the algorithm, and with any hope the second phase of the project will commense.

The second phase of the project involves implementing the algorithm on a massiveley parallel system, and hence the initial focus of this phase is the construction of a 32 node Beowulf cluster of parallela computers. The necessity for this construction is due to the large amount of CPU hours required to have the algorithm run continually. It would not be cost effective to run each OpenBrain instance on per CPU hour clusters such as Berkeley's own PSI cluster. Furthermore control of the hardware provides the opportunity to maximize the performance of each instance with respect to parallel processing power.

Once completed the cluster will be linked directly to a build system using hooks from Github which automatically pushes Erlang OTP code for the OpenBrain algorithm to every node. This will allow for easy prototyping and live code hot swapping.

Finally much research can be done in a variety of external environments using the universal intelligence indicator and other metrics that evaluate performance. Specifically, OpenBrain will be compared to Deep Reinforcement Learning. These applications and comparisons will be understood with more specificity as the project reaches its maturity.