A HIGH-SPEED ARCHITECTURE FOR BUILDING HYBRID MINDS

Oisín Mac Fhearaí and Mark Humphrys and Ray Walshe
School of Computing, Dublin City University, Dublin, Ireland
{omacfh, humphrys, Ray.Walshe}@computing.dcu.ie

Keywords: collective intelligence, agents, distributed problem solving

Abstract: A resurgence of interest has taken place supporting the idea of an intelligence composed of many simple components or "subminds". There is a growing consensus that, rather than a small number of "elegant" techniques for reasoning, inference or learning being the key to A.I., a model more likely to succeed might consist of perhaps thousands of simple agents co-operating such that an emergent intelligence is seen. The World-Wide Mind project is our attempt to facilitate this approach to scaling up artificial intelligence by enabling large hybrid systems to be built by multiple authors. The goal of the research described in this paper is to improve greatly the speed of the World-Wide Mind platform with a new communications protocol and implementation, to improve the user API and human interfaces, and to investigate methods of automatically constructing effective hybrid minds from the work of multiple authors, as well as to encourage collaboration between A.I. researchers worldwide.

1 BACKGROUND

A resurgence of interest has taken place supporting the idea of an intelligence composed of many simple components or "subminds" (Minsky, 1985; Bryson, 2007; Brooks, 1991). Classical approaches to solving the A.I. problem might be criticised for betting on single techniques to solve everything (e.g. neural networks, propositional logic, genetic algorithms, planners), when it seems that humans rely on a large, messy array of strategies and heuristics for cognitive function (Minsky, 1985). There is a growing consensus that, rather than a small number of "elegant" techniques for reasoning, inference or learning being the key to A.I., a model more likely to succeed might consist of many simple agents co-operating such that an emergent intelligence is seen.

1.1 The World-Wide Mind

We define a **world** as an instance of some problem - such as a chess game or a block-stacking puzzle - which may be solved by carrying out **actions** - such as placing a chess piece on the board, or lifting a block. At each timestep, the world produces a partial **state** representing the agent's sensory inputs. In a chess game, this state will be the complete state: the positions of all the remaining pieces on the board. In other worlds, the state will be a small subset of the complete state - for example, in a poker game, the contents of the opponents' hands may not be visible. This state is passed as input to a **mind**, which performs some computation and returns a suggested action to perform in the world. The pattern of interactions repeats until the problem is solved or the **run** is otherwise terminated (for example, a chess game ending in a tie).

This project attempts to make minds and worlds available as services distributed across the internet and encourage participation in the creation and use of these services, and to facilitate the construction of **hybrid** minds (Mind-Ms) which query subminds for suggested actions. Previous efforts in enabling large-scale multi-author

collaboration have not continued into widespread use in building complex hybrid minds, which is partly what this work intends to address.

2 RELATED WORK

There is a large body of work on the topic of modular agent architectures, such as the Society of Mind (Minsky, 1985), the subsumption architecture (Brooks, 1991) and CogAff (Sloman, 2002). There are also a number of implementations of distributed multi-agent systems that fulfill different requirements (e.g. encryption and fault tolerance in Cougaar (Helsinger et al., 2004) and agent mobility in aglets), but the extensive functionality and flexibility of these systems comes at a cost of some inaccessability for interested newcomers. The aim of these frameworks is to facilitate multi-agent applications such as logistics and online trading (Rogers et al., 2007), rather than explicitly to explore large-scale distributed cognition involving multiple authors. These frameworks provide architectures that make assumptions about how information should be represented and communicated, how systems should be organised and what their responsibilities are. It is our belief that fewer requirements and responsibilities placed upon mind authors may facilitate more participation.

A major goal of the W2M project is to minimise the barrier to entry for researchers, students and casual programmers to create and test their minds and worlds freely (Walshe et al., 2004). Authors are free to define the world state and action representations, as well as the interactions between sub-minds as they see fit. In our implementation we have selected a simple scheme defined by an XML-based markup language where minds return a suggested action in response to a *getaction* message containing the current state observable by the agent in the world, and worlds respond to *getstate* and *takeaction* methods.

It is noted in (Bryson, 2007) that none of the leading architectures documented in an earlier review of action selection work were still actively used seven years later. Perhaps this suggests that, rather than re-inventing architectures in an attempt to create intelligence, what might really be needed is simply a large set of agents implementing various behaviours and problem-solving strategies so that, while no one person need understand every element of the whole mind, it is collectively created by a large community.

Collaboration and sharing exists within artificial intelligence research; for example, there are websites which serve as repositories for machine learning code (Kantrowitz, 2009) and training datasets (Asuncion and Newman, 2009). These repositories are useful, but the steps required to install or adapt an existing solution differ in each instance and there is little consistency in the types of programs and interfaces provided. Users must download the code and often patch it to compile on their own machine, and must adapt the program to suit the interface and problem structure they wish to solve, if indeed the program addresses the chosen problem.

RoboCup (Visser and Burkhard, 2007) and the DARPA Challenge (Rouff, 2007) are closer to what we wish to achieve, but the problem domains are specific and there is no clear way to build and share hybrid minds. These projects focus on competition but not explicitly on re-use. Nevertheless the significant interest in the development of agents indicates that collaborative approaches to A.I. research can be successful.

There are limits on what one researcher or even one lab can do. If we succeed in building a fast, usable platform which can be used to build diverse problem worlds and create minds composed of a multitude of subminds - *if multi-author collaboration could be made easier* - then the interest generated and subsequent studies could have a positive impact on A.I. research.

3 WORLD-WIDE MIND "V2.0"

We outline here some of the recent changes that may realise the vision of the World-Wide Mind project and make it usable across the world.

3.1 Speed

The initial design of the platform (Humphrys, 2001) embodied minds and worlds as web services, operating over the internet by sending messages over the HTTP protocol. This enabled hybrid minds to be built and evaluated (Walshe et al., 2004), but was impractically slow for large hybrids. When the system was used as the basis for undergraduate assignments in third year A.I. courses, the majority of submitted minds were monolithic programs which did not seek the advice of other minds to select actions.

As the primary objective of the project is for the platform to be used widely by a large number

of authors, improving the latency and throughput of messages was a major concern.

3.1.1 Fast communication protocol

The implementation of minds and worlds as web services afforded simplicity and transparency in distributing minds and worlds online. However, there is a time penalty to be paid for the use of a web application server (Apache Tomcat) and the HTTP protocol to wrap messages.

To avoid these bottlenecks, the web application server was replaced by a custom TCP-based server which reads and writes XML messages across the network as length-prefixed strings. Similarly, while one goal of the platform is to be tolerant of errors in the generated XML, we have optimised the case where the XML is well-formed (since it will normally be machine-generated).

3.1.2 Taking advantage of centralisation

Although videos were hosted online before Youtube appeared, the act of hosting one's videos was not a simple task. There were issues with codecs, browser plugin versioning and so on. The arrival of Youtube caused an explosion in the amount of content placed online by users - in part because it removed these burdens of hosting and interoperability, but also because it served as a centralised, searchable directory of videos accessible from any machine with internet access.

We decided to host most of the minds and worlds ourselves - that is, to set up a "Youtube of A.I." to which people may upload minds and worlds, so that much re-use of other people's work would involve direct method calls, avoiding network access or parsing of XML. The resulting increase in speed is significant, especially when a mind queries other minds to assist in action-selection - since the queries are executed serially, passing messages through the TCP stack as XML would add a tangible latency to the process and penalise the use of many subminds.

To support this, a new type of message was devised: the *continuerun* message, when received by a mind service, causes it to carry out the run directly by taking actions in the world (to which the world responds by returning the newly-observed state, avoiding the need to send explicit *getstate* messages). As the mind carries out the run, it passes information on states seen and corresponding actions taken back to the client asynchronously. The client can see what happened during a run without slowing down significantly the communication between world and mind.

Note that while these measures are intended to greatly optimise speed in many cases, it may still be necessary to rely on world and mind services on the different servers. In this case, network and message encoding/decoding latencies add an unavoidable time penalty for each exchange of messages which is noticeable on long runs.

As a result, the speed at which a conversation is carried out between a mind or world service on different servers has increased by a factor of 100 (with roundtrip time from $\approx$1s down to 10ms) although this depends greatly on network latency and mind/world complexity. In cases when the mind and world are computationally simple and hosted on the same server, we have seen performance improvements of over a thousandfold.

3.2 User API

In the interests of ease-of-use and attractiveness to newcomers, an API was developed for interacting with the components of the system in a uniform manner, without needing to deal with the XML or network API. Interactions with remote minds or worlds are encapsulated by the `RemoteMind` and `RemoteWorld` classes.

3.3 Web interface

The initial implementation required the installation of a Java program locally before users could perform and view runs. To lower the entry barrier for users, an AJAX-based front-end which relies on an XML backend was created. The web interface allows anybody to start and examine runs, and to view a dynamic scoreboard for each world. Users can also register on the site and upload their own minds and worlds as JAR files, making them immediately ready to run (and re-use).

3.4 Visualisation of runs

A graphical viewer was created by Ciarán O'Leary for his implementation of Tyrrell's simulated animal world (Walshe et al., 2004), displaying at each timestep the two-dimensional grid with icons displaying locations of animals and other features in the world. This viewer was extended into a uniform user interface for world designers to render the world state visually. Functionality was added by Brian Monks to render runs as a series of images and as video.

3.5 Avoiding infinite loops

Since we can neither guarantee the correctness of user-supplied code nor reliably detect infinite loops, deadlocks and other difficult errors, a heuristic scheme was implemented which kills running services that take too long to respond to a request. This can and almost certainly will result in the termination of runs which were slow for genuine reasons (such as complex processing in a mind or world, or a mind which queries a large number of remote subminds), so the timeout threshold will be calibrated as more minds and worlds become available.

3.6 Testing the system

The resulting system was opened up for undergraduate computer science students tasked with writing Mind-Ms to control an animal in a modified Tyrrell world (Tyrrell, 1993). The requirement for students to write Mind-Ms resulted in a variety of different approaches, with some delegating to do-nothing minds which return a constant value in certain situations (e.g. "delegate to the Sleeper mind if we want to sleep"), while some called successful minds on the scoreboard (many of which were themselves Mind-Ms) and some used more complex strategies to select actions and minds.

3.7 Call graph

It can be informative to see the set of subminds called by an individual Mind-M, and in turn which minds are called by them. This is addressed by the addition of a call graph for each mind representing the set of minds it calls during each run. The server tracks messages to other minds and updates the call graph in a database.

4 RESULTS

An important question behind this work is whether a single, open platform can encourage productive collaboration on artificial intelligence work by multiple authors. To address this, we were able to use some of the minds created for the modified Tyrrell world by undergraduate students. Three non-hybrid minds were selected from a total of 117 for use in a hybrid mind, based on their names and the descriptions written by each author, with the intention of satisfying three goals: mating, sleeping in the animal's den and consuming food and water. Each of these subminds scored poorly when tested individually, with the best of the three (the mating mind) mating 11 times and living for 218 timesteps.

A hybrid was created which uses simple condition-rules to select which submind to obey at each timestep - e.g.: `if (mateNearby) return mater.getaction(state)`. This short program outperformed each of the subminds, mating 44 times in 3626 timesteps on its best run. We believe the hybrid was successful because the high-level behaviours specified by each mind collectively encapsulate a good strategy for mating and survival. The hybrid was able to select between these behaviours without explicitly encoding the low-level actions needed to satisfy each goal.

5 FUTURE WORK

5.1 Inviting collaboration

The platform itself will be of no use without problem worlds and minds to solve the problems. As the capabilities of the system grow, we will attempt to make the system available to a wider spectrum of users; and especially to artificial intelligence lecture courses and researchers worldwide. It is our hope that the platform will encourage users to collaborate by uploading minds to work together on problems such that an emergent intelligence is seen. Thus we hope to discover some novel methods of computation and digital cognition via combinations of minds using this system. Perhaps the most intelligent minds will make use of other minds which the Mind-M author does not (and need not) understand.

5.2 Communication between subminds

Although hand-coded minds usually do not give output other than a suggested action at each timestep, many A.I. algorithms do produce some form of data which could be used by hybrid minds, for example: discounted reward in reinforcement learning (Sutton and Barto, 1998), probabilities and degrees of truth/belief in Bayesian analysis and fuzzy logic, or heuristic solution cost in A*. This information could be passed back and forth between the hybrid mind

and its subminds and used to reason about which mind should be obeyed at a given moment, a task which is currently difficult (especially with subminds written by third parties) due to the lack of a unifying platform and API.

5.3 Constructing hybrid minds

So far we have not addressed the automatic construction of hybrid minds. To do so, we will carry out experiments to create hybrids based on statistical analysis of existing minds, ranking them by their performance at various goals which together constitute a holistic set of behaviours.

If we can break the problem into a series of needs to be satisfied, and if we can measure how well each need is satisfied by examining the states seen and actions taken during the run, then we can evaluate all available minds on how well they satisfy each chosen goal. For example, for the goal of minimising thirst in the Tyrrell world, we can evaluate a particular mind by recording the average value of the `perceivedWaterShortage` sensory input. We can then rank minds by their ability to satisfy the goal based on this metric. Once the minds which best satisfy each of the goals selected are identified, then implementing a hybrid mind could be a matter of specifying rules for the conditions under which each mind is called. Hybrid minds created with these techniques will be tested against each other (and against hand-built hybrids created by others) on the scoreboard.

6 CONCLUSION

There is currently no viable alternative platform which enables the easy construction of hybrids of other people's work on a global scale. We hope to make the platform so easy to use that it will attract authors of many diverse worlds and minds to deposit and test their programs on our site. Even if the project fails to find a significant number of users, it will shed light on future efforts and help point the way for further research in this field.

This research was funded by IRCSET's Embark Initiative.

REFERENCES

Asuncion, A. and Newman, D. (2009). UCI Machine Learning Repository. http://archive.ics.uci.edu/ml/.

Brooks, R. A. (1991). Intelligence without representation.

Bryson, J. J. (2007). Mechanisms of action selection: Introduction to the special issue. *Adaptive Behavior - Animals, Animats, Software Agents, Robots, Adaptive Systems*, 15(1):5–8.

Helsinger, A., Thome, M., and Wright, T. (2004). Cougaar: a scalable, distributed multi-agent architecture. In *SMC*, pages 1910–1917. IEEE.

Humphrys, M. (2001). The world-wide-mind: Draft proposal.

Kantrowitz, M. (2009). CMU Artificial Intelligence Repository. http://www.cs.cmu.edu/afs/cs/project/ai-repository/ai/0.html.

Minsky, M. (1985). *The Society of Mind*. Simon and Schuster.

Rogers, A., David, E., Jennings, N. R., and Schiff, J. (2007). The effects of proxy bidding and minimum bid increments within ebay auctions. *TWEB*, 1(2).

Rouff, C. A. (2007). Introduction: Darpa urban grand challenge editorial). *JACIC*, 4(12):1046–1046.

Sloman, A. (2002). How many separately evolved emotional beasties live within us?

Sutton, R. S. and Barto, A. G. (1998). Reinforcement learning: An introduction. *IEEE Transactions on Neural Networks*, 9(5):1054–1054.

Tyrrell, T. (1993). *Computational Mechanisms for Action Selection*. PhD thesis, University of Edinburgh.

Visser, U. and Burkhard, H.-D. (2007). RoboCup: 10 years of achievements and future challenges. *The AI Magazine*, 28(2):115–132.

Walshe, R., Humphrys, M., and O'Leary, C. (2004). Constructing complex brains: Building minds using sub-minds from biotechnology authors. In *1st IFIP Conference on Artificial Intelligence Applications and Innovations (AIAI-04), Toulouse, France, August 22-27, 2004, Proceedings*.