The use of Data Packets in a Behaviour Network to improve the Action Selection Mechanism

Brian Peach[1], **Peter Robinson**[2]
Department of Computer Science, University of Hull, Hull, United Kingdom
[1]b.peach@hull.ac.uk, [2]p.a.robinson@hull.ac.uk

Abstract

This paper presents a novel approach in the field of behaviour networks, using data packets to traverse a behaviour network enabling an agent to more accurately select appropriate behaviours. Behaviour networks are used as an action selection mechanism to allow agents to select the most appropriate behaviour in a given situation. Behaviour networks incorporate an energy spreading mechanism that allows it to determine which behaviours to execute. This research demonstrates that there are better methods for spreading activation energy around an action selection mechanism (ASM) and this is shown by implementing data packets to give an accurate selection of behaviours in a behaviour network.

Introduction

For many years' robots have been developed to solve a variety of tasks (Brooks 1986; Min & Cho 2010; Dorer 1999; Paikan et al. 2013; Petrick & Foster 2013). Each of these has applied a different control architecture to enable robotic performance of various tasks, but all have struggled to support flexibility in the face of dynamic environments. One mechanism which may help in this regard is the behaviour network, an action selection mechanism that can be used in control architectures (Lee & Cho 2014) to enable agents to select appropriate actions in changing situations.

A behaviour network typically consists of a directed graph representing the behaviours that an agent can perform and the restrictions upon them. Depending on the situation, considering both goals and sensor data, such a network will reactively select the best behaviour for the prevailing constraints, though with certain limitations (Tyrrell 1994).

This work aims to address these limitations with the introduction of quantization and recording of the energy used to evaluate the network. The idea is to send data packets through the behaviour network to transport the necessary energy from one behaviour to another. By embedding metadata into the data packets it is possible to solve the limitation outlined above.

The rest of this paper is organised as follows. Section 2 presents background and related work for behaviour networks. Section 3 describes the proposed data packet concept in detail. Section 4 shows the results of the experiments of the proposed solution. The Final section presents a summary and future works.

Background

In the early 90s Maes proposed the Agent Network Architecture (ANA) (Maes 1991c; Maes 1991b; Maes 1991a) as a planning mechanism to enable the selection of the most appropriate behaviour from a range of alternatives. This mechanism is based on the idea of a notional 'energy', which is allocated to a goal and spreads through a connection network to possible actions. Behaviours may contribute towards goals or inhibit their attainment; the combination of the network connectivity and the energy spreading mechanism determines which behaviour is preferred at any given time. This mechanism encapsulates a combination of reactive and planned behaviour selection, and is well suited to dynamic environments.

The behaviour network is defined by a collection of links amongst nodes representing behaviours, goals and the environment. Energy is added to the network by applying it to goal nodes or environment nodes, which encapsulate observations that the agents make of the environment. Figure 1 (Tyrrell 1994) shows the components of a behaviour node, and the types of link which connect to it. Each link allows the transfer of energy to or from other nodes.

Each behaviour contains a precondition list, consisting of a list of propositions that must be true in order for the behaviour to be executable. Behaviours also have an 'add list' of propositions that the behaviour makes true once it has been executed. Finally; the behaviours have a 'delete list' of propositions that the behaviour makes false once it has been executed.

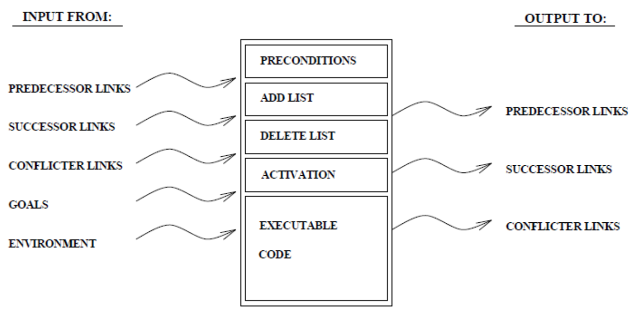

Figure 1. The components of a behaviour (Brooks 1986, Maes 1991, Tyrrell 1994)

The predecessor links in the network connect two behaviour nodes when; a proposition is false, the proposition is in the precondition list of node A and the proposition is in the add list of node B. So node B (once executed) can help node A become executable, will create an active predecessor link from node A to node B.

The successor links in the network connect two behaviour nodes when; a proposition is false, the proposition is in the add list of node A and the proposition is in the precondition list of node B. So node A (once executed) can help node B become executable, will create an active successor link from node A to node B.

A conflictor link in the network inhibits a behaviour node when; a proposition is true, the proposition is in the precondition list of node A and the proposition is in the delete list of node B. So node B will prevent node A from been executable and will create an active conflictor link from node A to node B.

The goal links in the network join behaviour nodes with goal nodes when; the activation energy in a goal is more than zero and goal Y is in the add list of node A. So node A is able to achieve goal Y.

The environment links in the network join behaviour nodes with environment nodes when; a proposition is true and the proposition is in the precondition list of node A. So node A is relevant to the given situation.

To begin the network evaluation process, the external nodes (goal and environment nodes) pass energy into the network according to formula defined rules (Tyrrell 1993; Tyrrell 1994). Once activation energy has been added to the behaviour nodes of the network from the external sources, then the energy is spread to identify the ideal action. The amount of energy that is spread from one node to another is a defined percentage of that node's activation energy. Each of the nodes in the behaviour network are connected by a combination of predecessor, successor and conflictor links. Predecessor and Successor links will increase the amount of energy in each node, while conflictor links will inhibit the spreading of the activation energy.

When the energy spreading mechanism is complete, the average energy of all nodes is normalised, and nodes with energy levels above a defined threshold are selected as candidates for activation. At any given time, the node with the greatest energy represents the preferred behaviour.

The original rules and formulas for each of the link types were reviewed and improved by (Tyrrell 1993; Tyrrell 1994) and then extended by (Decugis & Ferber 1998) who claimed to have solved some of the problems found in the original behaviour network (such as deadlock between multiple conflicting goals and improving reactivity between actions).

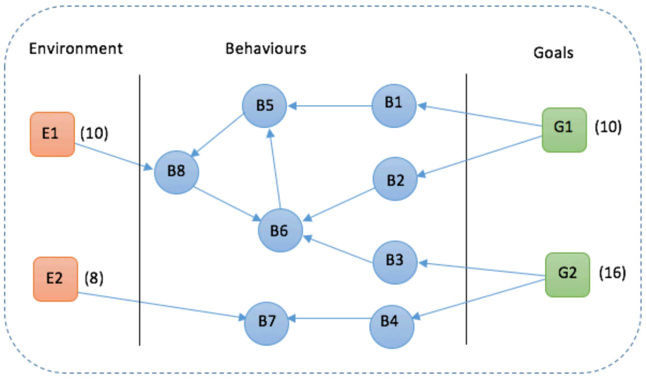

Figure 2. An example behaviour network

An example behaviour network is shown in Figure 2. It shows two goals and two environment nodes connected to eight different behaviours forming a behaviour network. A more complex, real world situation is described as a behaviour network in Figure 4.

The behaviour network was designed to work in dynamic environments and is therefore more reactive than traditional deliberative planning mechanisms. More recent work has been done to combine both these types to create a hybrid system (Lee & Cho 2014), They combined the behaviour network with a planning mechanism, to plan the sequence of actions that could also be adjusted according to exceptions and environmental changes.

Behaviour networks have not received a lot of interest in recent years as other techniques in AI have become more active and established in the field of controlling robot behaviour. The motivation for returning to the technique in this work stems from a larger project to assist robots in complex and dynamic environments.

During the implementation process of the behaviour network, some key issues were noted that have not been raised in any of the supporting text (Maes 1991a; Tyrrell 1993; Tyrrell 1994). These issues revolve around the process of spreading energy around the network. (Tyrrell 1993) explains that energy must first enter the network via the goal nodes and the environment nodes. The energy is passed between the nodes via the predecessor, successor and conflictor links. However; there is little discussion on the order in which the behaviour nodes should pass energy between themselves, and this leads to a variety of problems in more complex networks.

For any given network there may be many possible options for the order in which the energy should be spread. Depending on the order of evaluation of the behaviour nodes, for the energy spreading process, the energy allocated to specific behaviours after each iteration can vary. This inconsistent distribution of energy in the network, could be caused by selecting different start nodes, nodes not receiving all their energy before distributing their own and the possibilities of feedback loops between the behaviours.

Method

Each of the methods that were used during the implementation of a behaviour network adopted a general set of rules prior to selecting the order that the behaviours should spread the energy between themselves. The external nodes, such as the goals and environment nodes, would spread energy into the network first before the internal behaviours could spread energy. This was to ensure that there was sufficient energy in the network to begin with. Links augmenting other nodes' energy levels (predecessor and successor links) were selected before inhibition nodes (conflictor links) again to ensure that there was energy in the nodes prior to spreading the energy.

The early stages of implementation revealed a variety of different problems, relating to the mechanism of energy spreading and the consistency of its results. One such issue is possibility of a cyclical structure within the network (as shown in Figure 2, for example, prevents a clear definition of completion of energy spreading, since each node in the cycle passes energy to its successors, which in turn pass energy back. Such loops must be interrupted arbitrarily, and the resulting energy at each node depends upon the point of interruption in the cycle (which is not desired).

Even if a given network does not contain cycles, the results can depend on the order of evaluation of links. For example, energy can be spread from each connected behaviour in sequence following the links from each node in a depth-first search pattern, following the links until the end of the tree is reached and then back tracking to an unexplored node. This mechanism is simple, but has the flaw that the system does not know whether a behaviour has received all of its inputs before sending energy to the next node. This makes the resulting energy spread depend on the order of evaluation of the nodes in the network rather than its structure.

Further, if the energy spreading mechanism is applied iteratively, as Maes suggests, the resultant energy distribution varies with iteration, and does not necessarily converge. Depending on the number of iterations of the energy spreading mechanism, the behaviour network may select different behaviours to execute.

A variety of tests were performed on a behaviour network (Figure 2), starting with a random order for which to select behaviours and spread energy. A sample of the results from this test in shown in Table 1.

After 1 iteration	**After 10 iterations**	**After 100 iterations**	**After 1000 iterations**
B1 (0)	B1 (0)	B1 (0)	B1 (0)
B2 (0)	B2 (0)	B2 (0)	B2 (0)
B3 (0)	B3 (0)	B3 (0)	B3 (0)
B4 (0)	B4 (0)	B4 (0)	B4 (0)
B5 (3.59)	B5 (3.1)	B5 (3.07)	B5 (3.07)
B6 (2.93)	B6 (3.41)	B6 (3.44)	B6 (3.45)
B7 (2.89)	B7 (2.89)	B7 (2.89)	B7 (2.89)
B8 (0.00)	B8 (0.00)	B8 (0.00)	B8 (0.00)

Table 1. Results from using a random order of behaviour nodes.

The results in Table 1 differed each time the simulation was executed. Proving that; not only is order that the energy spread important, but also that an uneven distribution of energy exists in the network. An alternative method for selecting the order to spread the energy was then tested, based on a depth first approach, however; this then led into issues with potential feedback loops existing in the network. The previously suggested methods for spreading energy through the network are clearly inadequate for more complex networks, each leading to an undesirable distribution of energy in the behaviour network.

We propose a new mechanism to allow the behaviour network to evenly distribute energy around the network regardless of the order of evaluation or the number of iterations: the use of data packets.

Data Packets

The main problem of instability in the behaviour network occurs for two reasons. Firstly, for a node to correctly distribute energy to its successors, it must first have received its full complement from all its predecessors. Whether or not this has happened is hard to determine, and depends upon the order of evaluation of the nodes. Secondly, if the network contains cycles, whether or not a given node has received all its input energy remains undefined because some of the energy it emits will return as an input.

The proposed solution to both of these issues is to treat the energy as packets with associated metadata rather than simple values. When a behaviour needs to send its energy to the next behaviour it will create a packet to send (Figure 3). In addition to an energy value, each packet contains a list of all of the previous behaviours from which it has received energy, this permits a recipient behaviour to reject energy which has originated from itself, or if required to prevents it from sending more energy to a node which has already received it.

An additional modification to behaviour nodes accommodates the tracking of energy packets. Each behaviour now stores not only its current energy level, but also, a list of all of the data packets that it has received and a total energy value which is a sum of its energy and the energy of all of the packets that it contains.

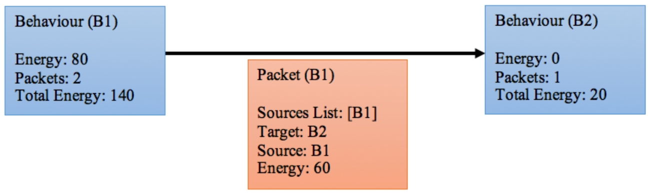

Figure 3. The contents of a behaviour and a packet

When a behaviour needs to send energy it will create a packet and put a proportion of its own energy into the packet (using a formula based on the link type) (Tyrrell 1994). The packet will then travel along the links of the same type (e.g. predecessor). When it arrives at a new behaviour the packet will be stored in that behaviours list of packages. The system will then check if the behaviour that packet arrived at has an outward link of the same type and if so, the behaviour will create another packet and send a proportion of energy (from the packet it just received) to the next behaviour. This process of creating and sending packets will continue until no packet can travel any further through the network.

This approach to labelling and tracking energy in packetized form allows much simpler and more consistent evaluation of the network. Because at any given time a node's energy is stored as a list of component packets rather than a single value, a node can spread only new energy each time it is received without the duplication inherent in maintaining only a single value. The order of evaluation of nodes ceased to be of importance; all that is required is that each node is evaluated once. Cycles in the network (which are valid logical constructions, but make evaluation difficult) also cease to be a problem; because each packet contains a list of the nodes which have contributed to its energy, it can be prevented from giving energy to a node from which it has already received it. There can therefore be no evaluation loops, and iteration becomes unnecessary.

Results

The data packet approach was tested on a basic behaviour network, containing a small number of arbitrary behaviours each connected via different types of links. Figure 4 shows this test scenario in a behaviour network. Here there are two goals a robot could have; ‘Explore’ will have the robot explore and learn about its environment and ‘Collect Yellow Blocks’ would have the robot find and move yellow blocks to a different location. The goal ‘Explore’ can be achieved with a behaviour ‘Move’, which will have the robot move in a random direction. The behaviours ‘Pick up Yellow Block’ and ‘Put down Yellow Block’ allow the robot to interact with a yellow block to achieve the goal of ‘Collect Yellow Blocks’. Finally, ‘Pick up Blue Block’ is included allow for conflict in the system; this is a desirable action but one which prevents the robot performing others. The situation of the environment is that the robot is in front of a yellow block which is ready to be picked up, the robot's hand is empty and it is too far from a blue block to interact with it without moving. Using these conditions, the appropriate links are used to connect the behaviours, shown in Figure 4.

The first implementation of the data packet approach followed the same approach as (Tyrrell 1994) where energy was sent into the network first via the goal and environment links. The predecessor links were then used to send energy around the network followed by successor links and concluding with inhibiting energy via the conflictor links. The results from this implementation is shown in Table 2.

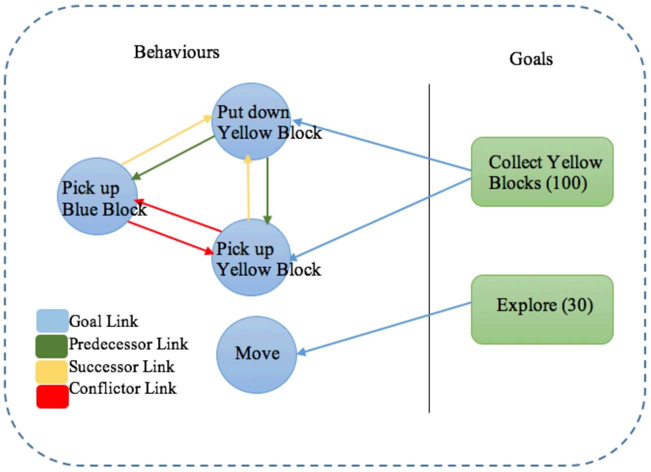

Figure 4. An example of a real world situation shown as a behaviour network.

Behaviour	Energy	Total Energy
Put down Yellow Block	0	60
Pick up Yellow Block	20	60
Move	30	30
Pick up Blue Block	0	27.5

Table 2. Results from using the packet approach with different types of links.

The results in Table 2 show that the agent does not have a clear choice for what it believes that the best action to take would be. The behaviours “Put down Yellow Block” and “Pick up Yellow Block” had equal highest energy even though the “Put down Yellow Block” behaviour cannot be executed given the situation of the robot. The reason for these results is the order in which the energy is passed via the links. The results in Table 2 followed the order; predecessor, successor and conflictor.

The literature on behaviour networks states, that once the energy spreading mechanism has concluded, the system should check for behaviours whose energy is greater than a global threshold. If there are no matches then the energy spreading mechanism should be repeated until a behaviour has surpassed the global threshold. It was then noted that there was a possibility for the “Move” behaviour to win out over the other behaviours, especially after multiple iterations of the energy spreading mechanism. This is because the “Move” behaviour is in a separate ‘system’ to the other behaviours, making it so it does not need to pass its energy to any other behaviour and that it will not be inhibited either. This is a problem if the other ‘systems’ in the behaviour network are what we would like the robot to do, more than exploring the environment. To solve this the amount of energy passed from the goal ‘Explore’ would need to be reduced or have some other inhibition setting that could be applied to it. The method that the behaviours use to pass energy to other behaviours is to send a proportion of energy based on its current energy level. This could explain the unsatisfactory results as when a behaviour sends energy to another behaviour via a predecessor link, it is reducing the amount of energy that behaviour has. When that same behaviour then has to send energy via a different link type it may find that its energy level is lower than expected or empty. For example; say behaviour 1 has an energy of 50 and it sends 40 energy to behaviour 2 via a predecessor link. Behaviour 1 then has 10 energy left for when it needs to send energy to behaviour 3 via a successor link. However; behaviour 1 could have also received a packet of energy from another behaviour making it so that behaviour 1 has 10 energy and a packet of 30 energy. The current solution of the system has the behaviour only send energy from its own source and not from any packets it may have received. This would work fine if there was only one type of link to consider but when multiple links are introduced the system needs to be modified.

The second implementation involved merging the energy in each of the packets with the remaining energy stored in each behaviour for each type of link. For example; following from the previous example; behaviour 1 has sent 40 of its 50 energy to behaviour 2 via a predecessor link and it has received a packet from another behaviour. Previously it would create a packet using the remaining 10 energy it has and send that via a successor link, instead; before it creates a new packet it will merge the energy in its current list of packets with the 10 energy it has remaining and send that proportion. This concept will show that the data packets can now distribute an even amount of energy around the network.

Behaviour	Energy
Put down Yellow Block	120
Move	30
Pick up Yellow Block	23.75
Pick up Blue Block	-8.75

Table 3. Results from using the packet approach with merging using different types of links.

Table 3 shows the results from using this approach with a link order of; predecessor, successor and conflictor. It shows that the ‘Pick up Blue Block’ has a negative value, this value is accurate as it is a behaviour that does not benefit the system. The ‘Put down Yellow Block’ has the most energy with 120, even though it is a behaviour that cannot be executed based on the current situation that the robot is in. Finally; the ‘Pick up Yellow Block’ has a value of 23.75, which again based on the current situation is incorrect. The order of the links was then changed to successor, predecessor and conflictor for the third implementation and the results are shown in Table 4.

Behaviour	Energy
Pick up Yellow Block	46.25
Move	30
Pick up Blue Block	13.75
Put down Yellow Block	0

Table 4. Results from using the packet approach with merging using different types of links.

Table 4 shows the results from using the packet approach with energy merging following the link order of; successor, predecessor and conflictor. Here the ‘Pick up Blue Block’ behaviour has a value of 13.75 which is low in comparison to the values in other behaviours. This means that this behaviour would be unlikely to be chosen for activation. The ‘Put down Yellow Block’ behaviour has a value of 0, which again is correct given the current situation the robot is in. Finally; the ‘Pick up Yellow Block’ behaviour has the most energy in that subsystem with a value of 46.25, making this the most likely behaviour to be selected. Overall these results show that the two executable behaviours were favoured over the non-executable behaviours in the system.

Conclusion

Behaviour networks are useful action selection mechanisms and are used in a variety of different control architectures. As documented there are potential flaws in the original text (Maes 1991a; Tyrrell 1994) whereby the order of the energy spreading mechanism greatly affects the outcome of the selected behaviour for execution. This work presents a solution to this problem by introducing data packets to traverse and distribute energy throughout a behaviour network. The data packets have been shown to be able to distribute energy throughout the network regardless of the order that the behaviours are selected to distribute energy in.

The results show that the data packet approach is the optimum method for spreading energy around a behaviour network accurately. The results show that the best method is to implement the energy merging technique and to follow the order of successor, predecessor and Conflictor for sending packets through a behaviour network.

The preliminary results presented in this paper form part of a larger project to develop a dynamic behaviour network. A system which can manipulate the behaviours in the behaviour network allowing an agent to work in complicated and unstructured environments. The future work will be to test the data packet methodology in more complicated and challenging scenarios. This work will then be implemented into the larger project of developing a dynamic behaviour network.

References

Brooks, R.A., 1986. A Robust Layered Control System For A Mobile Robot. IEEE Journal on Robotics and Automation, 2(1), pp.14–23.

Decugis, V. & Ferber, J., 1998. An extension of Maes' action selection mechanism for animats. From animals to animats 5: Proc. Fifth Int. …, (October), pp.1–8.

Dorer, K., 1999. Behavior networks for continuous domains using situation-dependent motivations. IJCAI International Joint Conference on Artificial Intelligence, 2, pp.1233–1238.

Lee, Y.S. & Cho, S.B., 2014. A hybrid system of hierarchical planning of behaviour selection networks for mobile robot control. International Journal of Advanced Robotic Systems, 11(1).

Maes, P., 1991a. A Bottom-up Mechanism For Behaviour Selection In An Artificial Creature. , pp.238–246.

Maes, P., 1991b. Learning to Coordinate Behaviours. Learning.

Maes, P., 1991c. The Agent Network Architecture (A N A). , 2(4), pp.115–120.

Min, H.J. & Cho, S.B., 2010. Adaptive behaviors of reactive mobile robot with Bayesian inference in nonstationary environments. Applied Intelligence, 33(3), pp.264–277.

Paikan, A., Metta, G. & Natale, L., 2013. A port-arbitrated mechanism for behavior selection in humanoid robotics. 2013 16th International Conference on Advanced Robotics, ICAR 2013.

Petrick, R.P. and Foster, M.E., 2013, June. Planning for Social Interaction in a Robot Bartender Domain. In ICAPS.

Tyrrell, T., 1994. An evaluation of Maes's bottom-up mechanism for behavior selection. Adaptive Behavior, 2(4), pp.307–348.

Tyrrell, T., 1993. Computational mechanisms for action selection. , pp.1–218.