4-15-2015

A Blackboard-style decision-making system for multi-tier craft control and its evaluation

Jeremy Straub

Hassan Reza
University of North Dakota, hassan.reza@UND.edu

A Blackboard-style decision-making system for multi-tier craft control and its evaluation

Jeremy Straub &Hassan Reza

Abstract

This article presents an approach for decision-making in support of the control of an autonomous system of multiple tiers of robots (e.g., satellite, aerial and ground) based on the Blackboard architectural style. Under the proposed approach, the system evaluates prospective approaches for goal satisfaction (identified by user selected final rules), identifies the lowest-cost solution and determines the best path to achieving the goal, via the analysis of the Blackboard rule and action set. Two different approaches to this rule and action path generation are discussed. This article presents the proposed Blackboard-style architecture for autonomous multi-tier control and describes its implementation. The benefits and drawbacks of the Blackboard-style approach are analysed, its extrapolation to the control of multiple heterogeneous craft is presented and the tradeoffs between the two approaches to rule-path generation are assessed.

Keywords: autonomous control, multi-tier control, blackboard architecture, robotic control

1. Introduction

The control of multiple robots with heterogeneous capabilities – both in terms of functionality and movement – is required for certain applications, such as planetary exploration, and beneficial for numerous others (e.g., persistent surveillance for defence and homeland security applications). The utility of these robots can be increased when they are able to collaborate, without direct controller involvement, to achieve their scientific or other goals. This autonomous control and collaboration, at a minimum, frees communications links to be used for data transmission (instead of control transmissions and the data required for decision-making). In some cases, it can provide synergistic value and allow faster performance of the mission objectives.

Prior work (Straub, 2012a; Straub & Fevig, 2012) has discussed how an autonomously controlled multi-tier system consisting of orbital, aerial and ground craft can be utilised to facilitate planetary exploration. The use of a Blackboard-style decision-making architecture (based on the Blackboard architecture developed by Hayes-Roth (Hayes-Roth, 1985)) has previously (Straub, 2013c) been discussed. This article presents a refined version of the concept presented in (Straub, 2013c) and discusses its implementation. The performance of two approaches for Blackboard solving in support of decision-making are presented. Finally, the extrapolation of this approach to distributed control is briefly discussed.

2. Background

The proposed approach draws from significant prior work in the two areas. First, prior work in the control of multiple heterogeneous (in terms of craft movement and other capabilities) craft is discussed. Then, prior work related to the Blackboard architecture is reviewed.

2.1 Control of multiple heterogeneous craft

Several approaches for the control of an autonomous system of multiple craft with heterogeneous capabilities have been proposed. Fink, Dohm, Tarbell, Hare, and Baker (2005) and Fink et al. (2007, 2011) have proposed a "tier-scalable" system where craft are controlled from a central decision making orbital node. This approach benefits from the comparatively greater computational resources available at this craft. However, the approach may yield suboptimal performance when regional conditions (for the various areas that system component craft are operating in) vary or communications are intermittent.

Corke, Peterson, and Rus (2005) looked at what is effectively a two-tier network. They utilised a network of sensors to guide an autonomous unmanned aerial vehicle (UAV). Their network can autonomously generate/re-generate paths and will also create transitional paths to guide the UAV from one path to the newly created one. The communications difficulties experienced in their field testing demonstrate a key problem with centralised control, network unreliability. Boano et al. (2010) characterised these communications problems and the resiliency of common protocols to them. They found that communications can be traded against latency, with a more latency-tolerant approach utilising only one-twentieth of the power of a less latency-tolerant one. Fairbairn, Bate, and Stankovic (2013) consider other impediments to system operations, concluding that environmental conditions pose the biggest threat to a system meeting its objectives.

De Poorter, Moerman, and Demeester (2009) consider inter-node communications in the context of designing sensornets with heterogeneous node capabilities. They concluded that autonomous selection of communications approaches is required to maximise system performance. An alternate mechanism for increasing system performance (considering the level of power draw imposed by radio communications) has been suggested by Tate, Bate, and Poulding (2008). They found that an optimal communications protocol could reduce energy usage by 45%, allowing this power to be devoted to other activities (or reducing the level of generation capabilities required, in the context of a planetary mission, reducing mission mass and launch costs). Reducing the level of data transmission needed by increasing the value of data (as discussed in (Straub, 2012b, 2012c, 2013a, 2013b, 2013d)) can be used in addition to this for further optimisation. An approach featuring approximation of data, suggested by Gupta and Uthra (2013), would similarly reduce transmission and thus power needs; however, the accuracy of (level of error in) the data produced by this approach is unknown to system users, making it unsuitable for many applications.

Colby and Tumer (2013) looked at the training of distributed multi-agent systems. They proffer that the use of "shaped rewards" in conjunction with a learning algorithm can produce as much as a two order-of-magnitude increase in system performance. Zhang and Lesser (2013) noted, however, that training can be problematic from a communications standpoint. They

proposed an approach to training that is robust with regards to coordination issues, requiring less bandwidth and increasing scalability. Hamzi, Koudil, Jamont, and Occello (2013) presented a prospective framework for controlling a network of nodes utilising a multi-agent approach; however, it lacks quantitative assessment leaving the approach's value unknown. Chun et al. (2005) and Haghighi (2014) both proposed the methods for making control decisions based on 764 2 J. Straub and H. Reza economic models; however, little attention is paid to the generalisation of these approaches, leaving their utility for other applications, similarly, unknown.

2.2 Blackboard architecture

Hayes-Roth (1985), presented the blackboard architecture based upon the Hearsay-II system (Erman, Hayes-Roth, Lesser, & Reddy, 1980) for robotic control. The Blackboard approach largely functions like an expert system (e.g., (Buchanan et al., 1983; Clancey, 1983; Feigenbaum, Buchanan, & Lederberg, 1970; O'Keefe, Balci, & Smith, 1986; Shortliffe et al., 1975; Waterman, 1986)); it differs in that it can trigger actions in addition to rules (expanding it from an recommendation generation system to a control system). The architecture is comprised of two blackboards: one is used for solving domain problems, while the other deals with control problems. These problems are solved via the invocation of rules on the blackboards (based on preconditions being met, placing it on the invokable list and its selection from this list). The initial paper (Hayes-Roth, 1985) failed to present any analytic data; however, Johnson and Hayes-Roth argued that applications such as the PROTEAN system (Johnson & Hayes-Roth, 1987) have demonstrated the architecture's efficacy.

Hayes-Roth's Blackboard system is not alone in solving problems in this way. Georgeff and Ingrand (1989) presented the similar SRI procedural reasoning system. This system, from their limited description, appears conceptually similar to the Blackboard approach, but not as robust (though this may be due to description and not system limitations). They note Hayes-Roth's paper (Hayes-Roth, 1985), published four years earlier; however, they utilise alternate terminology to describe the SRI system.

Several different enhancements to the architecture have been presented. Rice (1986), for example, developed Poligon, which is a language for implementing Blackboard problemsolving approach applications. It is designed to be used to encapsulate and represent the problem-solving behaviour of human experts. Le Mentec and Brunessaux (1992) created the Lisp-based Atome-tr system that is designed to produce Blackboard reactions in an acceptable timeframe for real-world control. They used a custom interrupt approach, dynamic planning and parallel processing to achieve this. Hewett and Hewett (1993) solved the problem that they identified while creating a common language for Blackboard solutions. It defines both syntax and mechanisms for maintaining the task list and for task activation.

A variety of different uses of the Blackboard architecture have been demonstrated. Brzykcy, Martinek, Meissner, and Skrzypczynski (2001), for example, used it for updating a perception network that is used for controlling autonomous robots. Fayek, Liscano, and Karam (1993) demonstrated its use for travelling through a passage and navigating across an irregularly

configured surface. Fox, Evans, Pearson, and Prescott (2012) consider how it can be used for mapping for robots with tactical sensing capabilities, while Yang, Tian, and Mei (2007) showed how the Blackboard architecture can facilitate learning. They used Q learning (a reinforcementstyle learning mechanism) to learn and share the learned behaviour for performing tasks. Shahbazian, Duquet, and Valin (1998) demonstrated the architecture's utility for a naval command system and a surveillance system.

de Campos and Monteiro de Macedo (1992) demonstrated, during the PANDORA project, how the Blackboard approach could be used to aid the autonomous navigation and control of vehicles in industrial and agricultural applications. Michael, Stump, and Mohta (2011) showed how it can be used for controlling micro-aerial vehicles, while Carroll, Boyd, and Denzinger (2008) demonstrated its use for controlling a group of UAVs and Goldin and Chesnokov (Goldin, 2011) demonstrated its use in controlling a spacecraft. Rubin, Smith, and Trajkovic Journal of Experimental & Theoretical Artificial Intelligence 7653 (2003) demonstrated how it can be utilised to detect prospective terrorist threats. Ronchi, Butera, Frascari, and Scaruffi (1987) and Rissland, Daniels, Rubinstein, and Skalak (1993) discussed how it can be used to help diagnose system problems. Tait, Schaefer, Hopgood, and Nolle (2005) showed how it can be used to detect problems with manufactured goods.

Numerous uses outside of robotics, for the Blackboard architecture, also exist. Plumbley et al. (2002) has demonstrated its utility for transcription. Dong, Shan, Ruan, Zhou, and Zuo (2013) have shown that it can help make apparel selections. McDonald, Gokhman, and Zachry (2012) have shown its utility for making inferences about social translucence, Wu, Peng, Zhu, and Zhang (2012) have demonstrated that it can be used for monitoring the elderly, Eldrandaly & Naguib (2013) demonstrated its effectiveness for aiding geographic information software selection, while Sanchez and Acquesta (2012) have even demonstrated its utility for risk management.

3. A Blackboard control system for multi-tier control

An approach to craft goal-setting was presented in (Straub, 2013c) is based on the Blackboard architecture. In this implementation, facts represent various pieces of information about the area of interest that are relevant to the decision-making process. Facts have multiple levels of scope. For example, a single fact (at the lowest level) may indicate a positive result for a particular test in a particular area. The next level up of fact (asserted, perhaps by a rule triggered because of the foregoing fact and other supporting details) might indicate that water existed at a given location. Finally, the detection of water at multiple locations in a region may support its classification as a dried up stream bed. Rules, thus, interconnect the facts in logically sound ways. Actions can trigger facts as well, but require a real-world or computational sequence to occur as part of the process (e.g., move to and collect data at a given location). Facts can be asserted by any tier craft and rules may combine facts which must be collected/asserted by craft of different tiers.

Under this approach, one or more rules are identified as final rules which, if triggered, indicate that the given goal has been met. Note that, under this approach, each identified final rule is taken to represent complete satisfaction of the goal (thus if multiple rules are required to

be triggered for completion, a rule that requires these rules as a prerequisite should be created and identified as a final rule). The identification of these final rules as well as the prerequisite relationships between rules (and the facts they assert/require) allows a network of rules to be created. This is shown in Figure 1.

Rules, however, are not the only part of a Blackboard architecture. Actions can be triggered; however, actions do not assert facts directly. Instead, they perform some task (e.g., triggering data collection) that may, indirectly, assert one or more facts. While some actions may have deterministic results, many (such as data collection activities) will not. Moreover, once an action is asserted, the Blackboard must continue operating with no immediate change in state (other than storing that the action has been triggered to prevent repetitive re-triggering of the same action). This more complex relationship is shown in Figure 2.

The Blackboard Solver is tasked with determining what to invoke during each cycle of Blackboard operations. The solver seeks to reach an end condition within a minimum amount of time and cost. It attempts to locate the lowest total-cost path, based on estimating costs for each rule or action to run and the value that is produced (in terms of moving the system towards an end condition). Two weight parameters are used to control the system's posture towards combined cost and value assessment. This, thus, can be determined by:

$$G_x = \frac{a \times V_x}{b \times C_x}, \qquad (1)$$

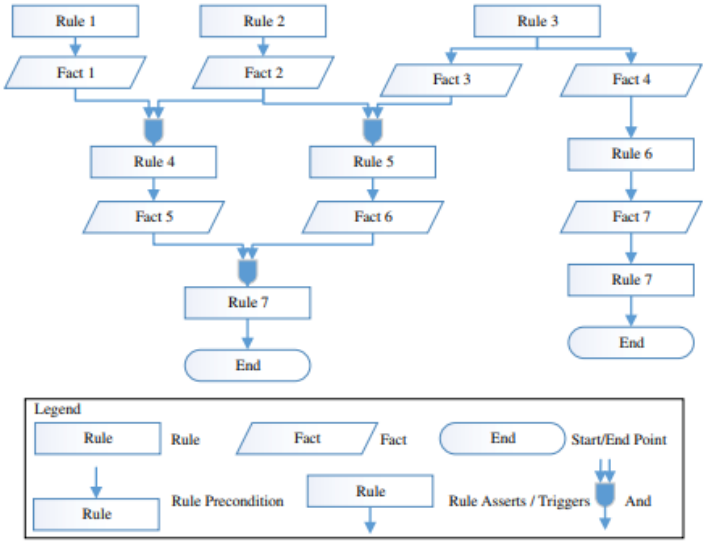

Figure 1. Network of rules showing end conditions and paths to arriving at these end conditions (modified from (Straub, 2013c)).

where a and b are the weight parameters and Vx and Cx are the value and cost of a given node. The selection metric, Gx, is a goodness value that is used for comparison purposes. As the cost of running a rule will generally be significantly (an order of magnitude or more) less than triggering

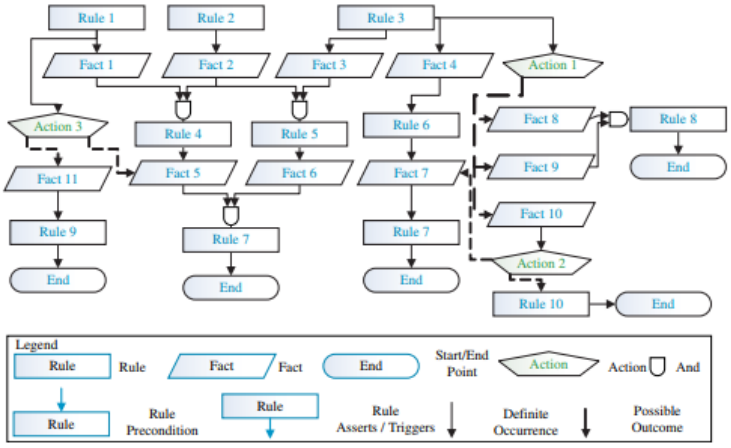

Figure 2. Addition of actions to the fact/rule network.

an action, rules that are invokable will generally be run before actions are triggered (however, if some rules are computationally intensive and thus have a higher cost, this may not always be true).

Several factors are considered in this cost and value determination process. Also, complexifying this is the fact that actions may have a single outcome with an associated uncertainty as to whether it occurs (which must be estimated) or multiple prospective outcomes (each with a given level of uncertainty). The cost of nodes (rules or actions) that are not currently invokable (i.e., which have unsatisfied prerequisites) is determined by their own cost plus the cost of the lowest cost projected way to satisfy their prerequisites. Figure 3 depicts the factors that are considered in the determination of the goodness metric for each node. The path with the lowest cost for the final rule is selected and the next nodes (that are part of the low-cost chain) are invoked.

During each operating cycle, the Blackboard, thus (presuming a case where high-cost rules are not present), looks for rules that are invokable and selects the best rule (based on the previously discussed criteria). Failing this, it will look for (higher-cost) actions that can generate data required for rules that are the next node in the lowest-cost chain and select the best action that is invokable. Once an action is invoked, the next cycle starts and (presuming no changes have occurred during this short period of time) another action to invoke may be identified. Parameters determine the number of actions that can be invoked at any given time and whether additional actions that will generate tasks for craft that have already received tasks from another action's invocation can be triggered. Figure 4 depicts this decision-making process.

In a change to the approach presented in (Straub, 2013c), final facts are selected for the work herein as it aided in the efficiency of system operations. The approach is functionally similar, as a rule that would previously have been a final rule can now simply assert a final fact. This change

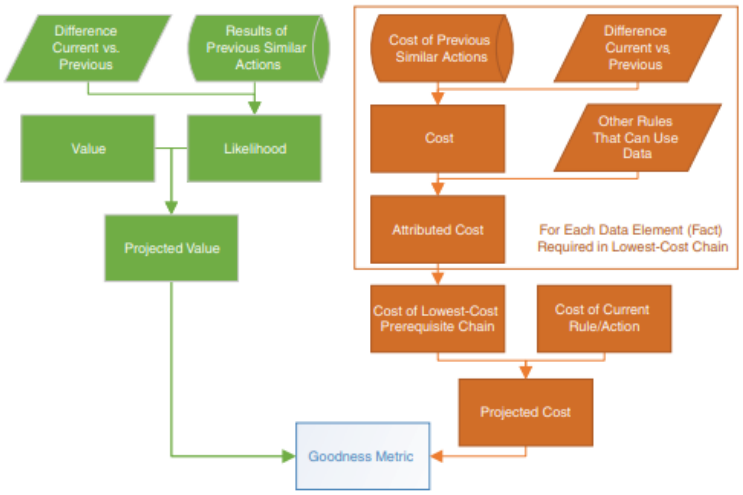

Figure 3. Determination of value, cost and goodness metric (modified from (Straub, 2013c)).

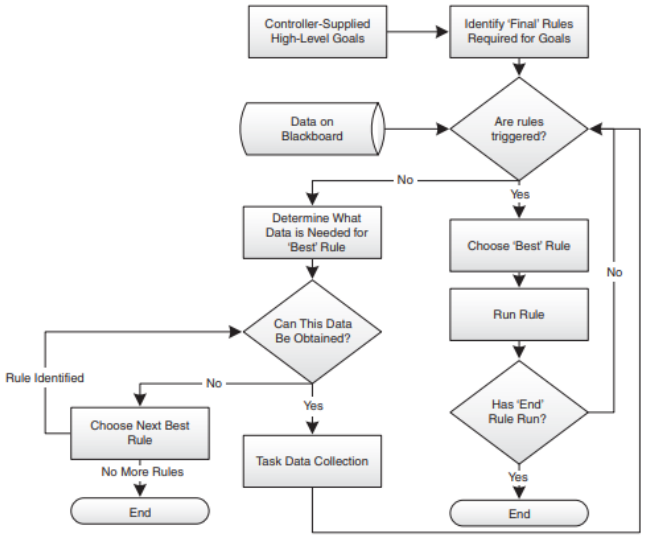

Figure 4. Diagram of system operations (modified from (Straub, 2013c)).

also facilitates actions being involved in the final aspect of solving a Blackboard problem, without requiring another rule evaluation.

4. Characterisation of performance and comparison of solver approaches

This section presents the results of comparative tests that were performed to assess the relative performance of a Blackboard-style best-path solver approach and a naïve solver, which operates in the manner described by Hayes-Roth (1985). First, a brief overview of the naïve solver is provided, for comparative purposes. Next, the methodology for the comparative testing is described. Then, the data collected is presented and discussed. Finally, analysis of this data is conducted.

4.1 Overview of naïve solver

The naïve solver works through operating the Blackboard's rules and actions in a forward (rule/ action execution) manner without prior knowledge of what rules and actions are part of the best path. It runs all rules and actions that are invokable (i.e., whose prerequisites are satisfied) based on their numerical order. Rules are invoked and mark applicable facts as asserted and trigger actions. Actions are invoked recursively, as an action may assert one or more facts and/or trigger Controller-Supplied High-Level Goals Identify 'Final' Rules Required for Goals Data on Blackboard Are rules triggered? Choose 'Best' Rule Determine What Data is Needed for 'Best' Rule Yes No Run Rule No Can This Data Be Obtained? Task Data Collection Yes Choose Next Best Rule No End Has 'End' Rule Run? End Yes Rule Identified No More Rules Figure 4. Diagram of system operations (modified from (Straub, 2013c)). Journal of Experimental & Theoretical Artificial Intelligence 7697 one or more additional actions. After each invocation, the Blackboard is checked to determine whether a final condition has occurred.

4.2 Experimental methodology

The Blackboard-based decision-making approach and naïve solver were implemented in a single application. This application creates a random network of rules, facts and actions (including random selection of pre-conditions and selection of initially asserted facts). A configurable number of rules, facts and actions can be utilised. For the testing conducted herein, 1000 facts, 1000 rules and 1000 actions were created. Approximately 30% of the facts were initially marked as asserted. For each rule, between 1 and 7 facts are randomly selected to serve as preconditions (both the number of facts utilised as preconditions and the actual facts are selected randomly). Between 0 and 7 facts are selected randomly (again, both the number and actual facts are randomly selected) to serve as assertions that are made when the rule is invoked. Actions, similarly have between 0 and 7 randomly selected (both the number and actual facts) facts as assertions that are made when the action is triggered. Each action is randomly assigned a cost of between 0 and 25.

One-hundred runs of the network generation and solving process were conducted. During each run, a new random network was created and a target node (representing a single final condition) was selected, randomly. The best-path solver was then used to identify the lowestcost path and the naïve solver was used to generate its path. The data collected during this process is presented in the subsequent section.

4.3 Data collected

The data collected by the process presented in Section 4.2 is presented in Table 1. This table presents the average values across the 100 runs conducted. The first two fields (solver iterations and path length) relate to the best-path solver and the latter two (number of rules and actions run) relate to the naïve solver.

The number of solver iterations value indicates the number of times that propagation routine had to process the blackboard's node-network to propagate the aggregate cost values through it. The solver continues running (iterating) until a run is completed without any changes being made. The path length is the number of executions that are required by the path identified by the best-path solver.

For the naïve solver, the number of both rules and actions run was tracked. The naïve solver stops when the user-designated final condition is identified as having occurred (or after a configurable number of runs without a solution being found. Out of the 100 test runs, four were excluded from the data presented in Table 1 because no solution was found (in this case the numbers of rules/actions run are significantly higher to the extent that it noticeably increases the average number of runs).

Table 1. Summary of collected data.

		# Run	
Solver iterations	Path length	Rules	Actions
8.28	9.13	1135.8	337.7

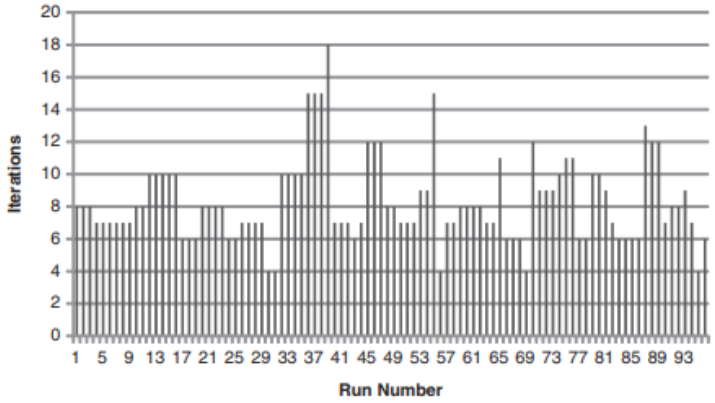

Figure 5. Solver iterations.

The data collected is also visually depicted. The graphs (Figures 5 – 7) show the level of variability across the trial runs. Figure 5 depicts the number of iterations of the solver run. Figure 6 shows the path length of the best-path solution. Finally, Figure 7 indicates the number of rule and action executions required by the naïve solver. There is very limited correlation between these three values across the set of trial runs.

4.4 Analysis of collected data

The collected data, presented in Table 1, requires additional details to assess. Two different metrics relevant to the analysis of this data exist. The first is the computational time required

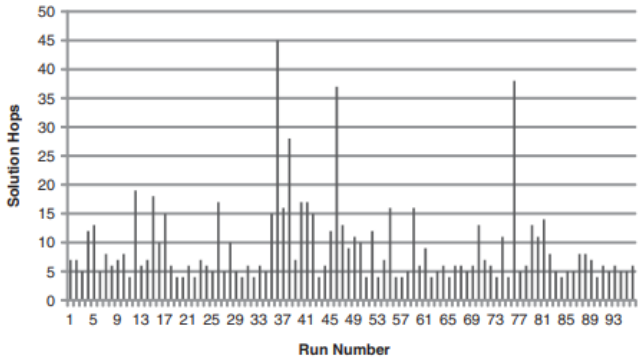

Figure 6. Solution hops.

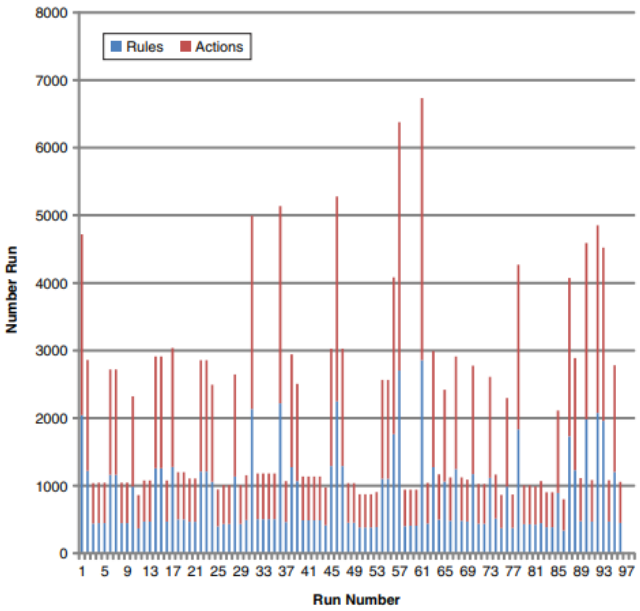

Figure 7. Number of rules and actions run.

and the second is the cost of actions run (rules only incur computational time and thus have no cost).

The average amount of time required for the 100 runs of the solver is presented in Table 2. An average of 8.3 iterations were run requiring an average of 1413.8 ticks each. The solver, thus, requires, on average, approximately 170.3 ticks per iteration.

Because of the recursive nature of action invocation, the time required to run a single action cannot be directly characterised. Instead, the duration (in ticks) is collected for running a set of actions and rules. This is presented in Table 3. Because the amount of computational time required for processing a single rule and action is not significantly different (compared with the ticks metric) it will be presumed that they require the same amount of time for the purposes of comparison (however, this is not always accurate).

Table 2. Characterisation of solver computational time requirements

Solver iterations	Time
8.3	1413.8

Table 3. Characterisation of time required for running actions and rules.

Rules	Actions	Time
850.5	1135.8	337.7

As shown in Table 3, an average of 850.5 rules and 1135.8 actions (1986.3 executions per run) were run during each of the 100 trials. This required 337.7 ticks, on average. From this, it is computed that each rule or action requires 0.17 ticks to run, on average.

Using this data, the average computational performance of the two can be assessed. Comparing the time values in Tables 2 and 3, it is clear that the naïve solver is much faster than the best path solver. The best-path solver takes, on average, 4.19 times as long to find a path as the naïve solver. It is, thus, considerably lower in computational cost. The impact of the lower computational cost, however, is relatively small compared with the difference in the number of actions that are must be executed. The naïve solver executes, on average 1135.8 actions while the best-path approach executes below 10, on average. The exact cost of these actions requires knowledge of the particular application, environment and task set that the software is being used in; however, given that many actions will involve robot movement and interaction with the real world, it will be significantly slower. Using the arbitrary (and overly optimistic) average value of 12.5, the naïve approach requires ten times as long (and this is presuming that actions take on average only 12.5 ticks – in the real world this could be orders of magnitude higher). Future work will assess the real-world, application-specific significance of these differences.

5. Comparing multiple naïve solver runs to single run

Because of the faster computational speed of the naïve solver, it can be run (on average) four times during the time that the best-path solver takes. The value of multiple runs is now considered. Table 4 indicates where the best performing run was found (by count) in each of the trials run. A total of 150 trials were run; 40 of them were discarded because one or more of the naïve solver runs completed without finding a solution (which produces a very high number of rule/action runs which impacts the statistics).

The value of using the lowest-cost (of the four executions) is characterised in Table 5. The reduction in the number of rules run saves (as per the previous characterisation in Section 4.4)

Table 4. Location of best performing run for rules and actions.

	Rules				Actions			
Best found	1	2	3	4	1	2	3	4
Number times	35	30	27	20	34	34	25	18

Table 5. Amount of executions saved by using best number of rules and actions versus first run.

	Rules	Actions
Average less	273.9545	370.8636

approximately 46.57 ticks. Based on the previously used arbitrary value of 12.5 ticks used for activation duration, the reduced number of actions saves 4635.8 ticks. Again, depending on the nature of the action performed, this could be a dramatic underestimate (12.5 ticks might be realistic for running a sub-routine, etc. but would dramatically underestimate any physical action; again, this must be assessed in context).

6. Discussion of the efficacy of the Blackboard approach and extrapolation to a distributed Blackboard

The work presented has shown how the network of Blackboard rules, facts and actions can be utilised to formulate plans for robotic decision-making, in support of mission goal completion. Other approaches that could be utilised include teleoperation (where controllers command the robot on specific actions to perform and figure out what actions best support goals themselves), scripted (or swarm-style) exploration where particular (perhaps coverage-optimised) paths are taken to explore a region and other decision-making network-based approaches (e.g., an expert system that generates suggestions). The ability to ascertain what is required to satisfy a goal provides the Blackboard-style (and to a lesser extent the expert system) approach with a significant benefit when compared with scripted or swarm-style exploration: it allows decisionmaking to avoid collecting unneeded or redundant data. Compared with teleoperation, the wait time required for transmission to a human, human review, human command decision-making and transmission back is eliminated. The expert system-style approach provides part of the benefit of the Blackboard styles (as the Blackboard architecture is an extension of an expert system); however, this approach must still find another way to turn suggestions into executed commands. Thus while similar performance may be attainable, starting from the expert system architecture is (likely needlessly) “reinventing the wheel” already demonstrated by the Blackboard-style architecture.

The extrapolation of this work to a distributed Blackboard is a subject for future work; however, this will necessitate solving a rule network that spans multiple Blackboards. To minimise communications requirements, the network of each node’s Blackboard is not shared with other nodes. The cost of sharing this data and the value of making a more informed decision must be compared with the (potentially sub-optimal) decision that would be made treating each Blackboard’s network as a black box (or requiring each to perform its own solving within its scope of influence).

7. Conclusions and future work

This article presented a Blackboard-style control architecture for a collection of heterogeneous autonomous craft. It described the implementation of this system, including how rules and actions are selected for invocation based on a cost/value analysis process. The performance of two different approaches to finding the path that will be used for valuing cost and value of rules

for selection purposes has been evaluated and the tradeoffs (faster computational performance versus faster real-world performance) assessed.

Future work will include adapting the proposed approach to serve as a distributed control system, expanding it to allow delegation of selected goals as assigned goals to subordinate craft. This more complex system will allow the comparison of a distributed control approach to a centralised one (such as the approach proposed by Fink et al. (2005, 2007, 2011)). It will also facilitate the comparison of the Blackboard versus non-Blackboard control strategies for this distributed control and the assessment of different approaches for the implementation of components of these control architectures.

Funding

This work has been supported by a Grant-In-Aid of Research from Sigma Xi, The Scientific Research Society, North Dakota EPSCoR (NSF # EPS-814442) and a Summer Doctoral Fellowship from University of North Dakota School of Graduate Studies. Facilities and equipment utilised in this work have been provided by the University of North Dakota Department of Computer Science and North Dakota EPSCoR (NSF # EPS-814442).

Disclosure statement

No potential conflict of interest was reported by the authors.

References

Boano, C. A., Voigt, T., Tsiftes, N., Mottola, L., Ro¨mer, K., & Zu´niga, M. A. (2010). Making sensornet MAC protocols robust against interference. *In Wireless sensor networks* (pp. 272– 288). New York, NY: Springer.

Brzykcy, G., Martinek, J., Meissner, A., & Skrzypczynski, P. (2001). Multi-agent blackboard architecture for a mobile robot. In *Proceedings, 2001 IEEE/RSJ international conference on intelligent robots and systems, 2001* (Vol. 4, pp. 2369– 2374). New York, NY: IEEE.

Buchanan, B. G., Barstow, D., Bechtal, R., Bennett, J., Clancey, W., Kulikowski, C., ... Waterman, D. A. (1983). Constructing an expert system. *Building Expert Systems, 50,* 127– 167.

Carroll, S., Boyd, J. E., & Denzinger, J. (2008). Data-centered control of cooperating UAVs: Flying airplanes with a multimedia database. Retrieved from http://citeseerx.ist.psu.edu/viewdoc/summary?doi ¼ 10.1.1.99.2334

Chun, B. N., Buonadonna, P., AuYoung, A., Ng, C., Parkes, D. C., Shneidman, J., ... Vahdat, A. (2005). Mirage: A microeconomic resource allocation system for sensornet testbeds. In S. Jha (Ed.), *The second IEEE workshop on embedded networked sensors: IEEE EmNetS-II,* May 30 – 31, 2005, Sydney, Australia (pp. 19 – 28). Piscataway, N.J.: IEEE.

Clancey, W. J. (1983). The epistemology of a rule-based expert system—a framework for explanation. *Artificial Intelligence, 20,* 215– 251.

Colby, M., & Tumer, K. (2013). Multiagent reinforcement learning in a distributed sensor network with indirect feedback. In *Proceedings of the 2013 international conference on autonomous agents and multi-agent systems*, Saint Paul, MN, USA (pp. 941– 948). Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems.

Corke, P., Peterson, R., & Rus, D. (2005). Networked robots: Flying robot navigation using a sensor net. In *Robotics research* (pp. 234– 243). New York, NY: Springer.

de Campos, A. M., & Monteiro de Macedo, M. (1992). A blackboard architecture for perception planning in autonomous vehicles. In *Proceedings of the 1992 international conference on industrial electronics, control, instrumentation, and automation, 1992. Power electronics and motion control*, San Diego, CA (pp. 826– 831). New York, NY: IEEE.

De Poorter, E., Moerman, I., & Demeester, P. (2009). An information driven sensornet architecture. In SENSORCOMM'09. *Third international conference on sensor technologies and applications, 2009*, Athens, Glyfada, Greece (pp. 553 –561). New York, NY: IEEE.

Dong, A. H., Shan, D., Ruan, Z., Zhou, L. Y., & Zuo, F. (2013). The design and implementation of an intelligent apparel recommend expert system. *Mathematical Problems in Engineering*, 2013, 1– 8. Article ID 343171. doi:10.1155/2013/343171

Eldrandaly, K., & Naguib, S. (2013). A knowledge-based system for GIS software selection. *International Arab Journal of Information Technology*, 10, 152–159.

Erman, L. D., Hayes-Roth, F., Lesser, V. R., & Reddy, D. R. (1980). The hearsay-II speech-understanding system: Integrating knowledge to resolve uncertainty. *ACM Computing Surveys (CSUR)*, 12, 213– 253.

Fairbairn, M. L., Bate, I., & Stankovic, J. A. (2013). Improving the dependability of sensornets. In *2013 IEEE international conference on distributed computing in sensor systems*, Cambridge, MA. New York, NY: IEEE.

Fayek, R. E., Liscano, R., & Karam, G. M. (1993). A system architecture for a mobile robot based on activities and a blackboard control unit. In *Proceedings, 1993 IEEE international conference on robotics and automation, 1993*, Atlanta, GA (pp. 267– 274). New York, NY: IEEE.

Feigenbaum, E. A., Buchanan, B. G., & Lederberg, J. (1970). *On generality and problem solving: A case study using the DENDRAL program* (Technical Report). Stanford, CA, USA: Stanford University.

Fink, W., Dohm, J. M., Tarbell, M. A., Hare, T. M., & Baker, V. R. (2005). Next-generation robotic planetary reconnaissance missions: A paradigm shift. *Planetary and Space Science*, 53, 1419– 1426.

Fink, W., Dohm, J. M., Tarbell, M. A., Hare, T. M., Baker, V. R., Schulze-Makuch, D., ... Miyamoto, H. (2007). Tier-scalable reconnaissance missions for the autonomous

exploration of planetary bodies. In *Aerospace Conference, 2007 IEEE*, Big Sky, MT (pp. 1 –10). New York, NY: IEEE.

Fink, W., Tarbell, M. A., Furfaro, R., Powers, L., Kargel, J. S., Baker, V. R., & Lunine, J. (2011). Robotic test bed for autonomous surface exploration of titan, mars, and other planetary bodies. *In Aerospace Conference, 2011 IEEE* (pp. 1 – 11). doi:10.1109/AERO.2011.5747267

Fox, C. W., Evans, M. H., Pearson, M. J., & Prescott, T. J. (2012). Towards hierarchical blackboard mapping on a whiskered robot. *Robotics and Autonomous Systems*, 60, 1356–1366.

Georgeff, M. P., & Ingrand, F. F. (1989). Monitoring and control of spacecraft systems using procedural reasoning. In *Third annual workshop on space operations automation and robotics* (pp. 209– 217). Washington, D.C.: National Aeronautics and Space Administration.

Goldin, D. (2011). Features of informational control complex of autonomous spacecraft. *In IFAC workshop "Aerospace guidance, navigation and flight control systems"*. Retrieved from http://lib.physcon.ru/file?id ¼ efc54562aea4

Gupta, A., & Uthra, R. A. (2013). Cluster based approximate data collection in wireless sensor network. *International Journal of Electronics and Computer Science Engineering*, 2, 740– 744.

Haghighi, M. (2014). Market-based resource allocation for energy-efficient execution of multiple concurrent applications in wireless sensor networks. In *Mobile, ubiquitous, and intelligent computing* (pp. 173– 178). New York, NY: Springer.

Hamzi, A., Koudil, M., Jamont, J., & Occello, M. (2013). Multi-agent architecture for the design of WSN applications. *Wireless Sensor Network*, 5, 14 – 25. doi:10.4236/wsn.2013.52003

Hayes-Roth, B. (1985). A blackboard architecture for control. *Artificial Intelligence*, 26, 251–321.

Hewett, M., & Hewett, R. (1993). A language and architecture for efficient blackboard systems. In *Proceedings, ninth conference on artificial intelligence for applications, 1993*, Orlando, FL (pp. 34 –40). New York, NY: IEEE.

Johnson, Jr., M. V., & Hayes-Roth, B. (1987). *Integrating diverse reasoning methods in the BBP blackboard control architecture*. In *Sixth national conference on artificial intelligence,* Seattle, WA. Menlo Park, CA: Association for the Advancement of Artificial Intelligence.

Le Mentec, J., & Brunessaux, S. (1992). Improving reactivity in a blackboard architecture with parallelism and interruptions. In *Proceedings of the 10th European conference on artificial intelligence* (pp. 255–256). New York, NY: John Wiley & Sons.

McDonald, D. W., Gokhman, S., & Zachry, M. (2012). Building for social translucence: A domain analysis and prototype system. *In Proceedings of the ACM 2012 conference on computer supported cooperative work* (pp. 637– 646). New York, NY: Association for Computing Machinery.

Michael, N., Stump, E., & Mohta, K. (2011). Persistent surveillance with a team of mavs. In *IEEE/RSJ international conference on intelligent robots and systems (IROS),* 2011 (pp. 2708– 2714). New York, NY: IEEE.

O'Keefe, R. M., Balci, O., & Smith, E. P. (1986). *Validation of expert system performance* (Technical Report). Blacksburg, VA, USA: Department of Computer Science, Virginia Polytechnic Institute and State University.

Plumbley, M. D., Abdallah, S. A., Bello, J. P., Davies, M. E., Monti, G., & Sandler, M. B. (2002). Automatic music transcription and audio source separation. *Cybernetics and Systems, 33,* 603– 627.

Rice, J. P. (1986). *Poligon: A systems for parallel problem solving* (No. KSL 86-19). Stanford, CA: Knowledge Systems Laboratory, Stanford University.

Rissland, E. L., Daniels, J. J., Rubinstein, Z. B., & Skalak, D. B. (1993). Case-based diagnostic analysis in a blackboard architecture. In *Proceedings of the national conference on artificial intelligence* (pp. 66 –66). Menlo Park, CA: AAAI Press.

Ronchi, J., Butera, G., Frascari, E., & Scaruffi, P. (1987). A distributed blackboard-based architecture for tele-diagnosis. *Artificial Intelligence for Engineering, Design, Analysis and Manufacturing,* 1, 103– 108.

Rubin, S. H., Smith, M. H., & Trajkovic, L. (2003). A blackboard architecture for countering terrorism. In *IEEE international conference on systems, man and cybernetics, 2003* (Vol. 2, pp. 1550– 1553). New York, NY: IEEE.

Sanchez, E. Y., & Acquesta, A. A. (2012). CRISIS: A system for risk management. *Systems,* 1, 3 – 26.

Shahbazian, E., Duquet, J., & Valin, P. (1998). A blackboard architecture for incremental implementation of data fusion applications. In *International conference on multisource-multisensor information fusion*, July 6 – 9, 1998, (pp. 455– 461). Athens, GA, USA: C.S.R.E.A. Press.

Shortliffe, E. H., Davis, R., Axline, S. G., Buchanan, B. G., Green, C. C., & Cohen, S. N. (1975). Computerbased consultations in clinical therapeutics: Explanation and rule acquisition capabilities of the MYCIN system. *Computers and Biomedical Research,* 8, 303– 320.

Straub, J. (2012a). *Multi-tier exploration: An architecture for dramatically increasing mission ROI.* Proceedings of the AIAA Space 2012 Conference, Pasadena, CA, USA.

Straub, J. (2012b). Model based data transmission: Analysis of link budget requirement reduction. *Communications and Network,* 4, 278– 287.

Straub, J. (2012c). Reducing link budget requirements with model-based transmission reduction techniques. In *Proceedings of the 26th AIAA/USU conference on small satellites*. Logan, UT: Utah State University.

Straub, J. (2013a). *Integrating model-based transmission reduction into a multi-tier architecture*. Proceedings of the 2013 IEEE Aerospace Conference, Big Sky, MT, USA.

Straub, J. (2013b). Application of model-based data transmission techniques to gravitational model data. *Journal of Data Analysis and Information Processing*, 1, 46 – 57.

Straub, J. (2013c). A data collection decision-making framework for a multi-tier collaboration of heterogeneous orbital, aerial and ground craft. In *Proceedings of the SPIE defense, sensing þ security conference,* Baltimore, MD, USA. Bellingham, WA, USA: SPIE.

Straub, J. (2013d). Fusion of data from multiple sensors with model-based data analysis. In *Proceedings of the SPIE defense, sensing + security,* Baltimore, MD, USA. Bellingham, WA, USA: SPIE.

Straub, J., & Fevig, R. (2012). *Multi-tier planetary exploration: A new autonomous control paradigm*. Proceedings of the AIAA Space 2012 Conference, Pasadena, CA, USA.

Tait, R. J., Schaefer, G., Hopgood, A. A., & Nolle, L. (2005). Defect detection using a distributed blackboard architecture. In *Proceedings of the 19th european conference on modelling and simulation,* Riga, Latvia. Pontypridd, United Kingdom: European Council for Modelling and Simulation.

Tate, J., Bate, I., & Poulding, S. (2008). Tuning protocols to improve the energy efficiency of sensornets. In *Proceedings of the fourth UK embedded forum,* Southampton, UK. Stevenage, Herts, UK: IET.

Waterman, D. (1986). *A guide to expert systems*. Boston, MA, USA: Addison Wesley Publishing Company.

Wu, P., Peng, H., Zhu, J., & Zhang, Y. (2012). Senscare: Semi-automatic activity summarization system for elderly care. In *Mobile computing, applications, and services* (pp. 1 – 19). New York, NY: Springer.

Yang, Y., Tian, Y., & Mei, H. (2007). Cooperative Q learning based on blackboard architecture. In *International conference on computational intelligence and security workshops*, Harbin, Heilongjiang, China (pp. 224– 227). New York, NY: IEEE.

Zhang, C., & Lesser, V. (2013). Coordinating multi-agent reinforcement learning with limited communication. In *Proceedings of the 2013 international conference on autonomous agents and multi-agent systems,* St. Paul, MN, USA (pp. 1101– 1108). Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems.