An Active-Logic Based Agent's Reasoning for Avoiding Futile Action Repetition

Anthony Herron
Dept. of Computer Science
Bowie State University
Bowie, United States
herrona0814@students.bowiestate.edu

Darsana P. Josyula
Dept. of Computer Science
Bowie State University
Bowie, United States
darsana@cs.umd.edu

***Abstract*—An agent working on tasks with tight deadlines must be cognizant of the passage of time and perform effective actions that would let it complete its tasks on time. Avoiding actions that do not contribute towards task completion is desirable; executing actions whose post-conditions are already met might not help in completing the task at hand. Moreover, repeating an action after post-conditions become true can be a wasted effort that affects the ability of the agent to complete its tasks. In this paper, we explore several sets of axioms that can be used by a time-situated agent for avoiding repeated actions. We also examine how the agent's knowledge temporally evolves while using these axioms for a time-constrained task and illustrate how these axioms affect task performance on a target search task.**

***Index Terms*—active logic, action-selection axioms, agent reasoning.**

I. Introduction

Over the span of an agent's lifetime, its knowledge changes based on its actions and perceptions. These changes, in turn, impact the decisions of the agent, influencing future actions and outcomes. When agents have a repertoire of actions at their disposal, there is a decision involved regarding which of the possible actions should be done and which should be avoided. Knowing whether an action has been previously tried and whether the post-conditions of the action hold true can help the agent decide whether to redo that action.

All actions whose preconditions are met are possible actions that can be executed, but not all of those actions will lead the agent toward a goal state. Particularly, performing the same action that has already been tried may not lead to a different outcome. A smart agent would take actions that steer it towards its goals, which in many cases would be the actions whose post-conditions are not met and are related to the goal. After executing an action once, if the post-conditions are true, a smart agent would avoid said action as constantly repeating the same action that has the post-conditions already met may not take the agent closer to its goal.

Besides, doing and redoing actions takes time; the clock is still running when performing futile actions. When agents work on tasks with tight deadlines, time is a limited resource that the agent has to manage carefully using the knowledge that it has at its disposal wisely. For an agent to be aware of this passage of time, it should be situated in time [1]. While being cognizant of the deadline for a task, the agent must choose and perform actions in a way that will lead to task completion. It would not make much sense for an agent to needlessly perform actions that have been done already since these repeated actions will generally waste precious time.

For instance, while searching for a car key when one is rushing to leave for the airport, time is limited as there is a deadline to meet (to reach the airport), and there are many areas to cover during the search. The goal is then to perform actions that will allow the agent to cover the areas where the key will likely be present quickly. Searching in places where the key doesn't exist affects the time it takes to find the key. Repeatedly searching a location where the agent already searched and ascertained that the key does not exist, instead of searching a new location, is futile and would consume valuable time left to complete the task. So, it is important to avoid such repeated actions that are not going to take the agent closer to completing the task.

In the search example, there is a better chance of finding a key in a spot that one hasn't searched previously. Even then, sometimes one has to retrace the areas that one has already searched in order to search in locations beyond the areas already searched or to get a better look at the areas previously searched. Hence, an agent that strictly avoids any repeated action is not desirable. If the entire area has already been searched and the target is not found, a wise option is to search the areas that have already been checked again. This would address the issue of simply overlooking the item, as in the keys being underneath something.

The conditions that lead to an action being selected or not influence agent behavior and task performance. If a successful post-condition prevents an action from being selected, an agent may never revisit a location that it has already been before. On the other hand, without a post-condition check, the agent may mindlessly keep repeating actions. Not only that, the amount of knowledge that is noted while the conditions are checked affects the choice of actions selected and hence agent behavior and task performance.

In this paper, we study the complex interactions in time between action-selection decisions and knowledge-condition checks using a time-situated agent based on active logic [2] in the context of reducing unnecessary repeated actions. We present six sets of axioms for action selection that an active-logic based agent utilizes to avoid repeated actions. Utilizing these sets of axioms in a search task setting with a deadline, we discuss the temporal interactions between action selection and knowledge-condition checks.

The following sections are organized as follows: Section II discusses the relevant literature in this area of research. Section III discusses the characteristics and functioning of active logic. Section IV explores six different action-selection axiom sets used in this work and how they avoid repeated actions. Section V details the experiment conducted, highlighting the different variables considered. Section VI examines the results from using the axiom sets during a search task.

II. Related Work

The reduction of redundant actions has previously been explored in the pursuit of creating intelligent agents. Chrpa, McCluskey, and Osborne [3] discussed techniques for determining whether an action is or is not redundant based on action dependencies. This work could not determine every possible redundant action within a plan as they are not well defined in the plan.

The work of Balyo [4] sought a similar goal but also focused on attempting to eliminate redundant actions from plans. The methods discussed covered solutions such as greedy algorithms and action reduction. The resulting elimination solutions were slow in some cases but were successful in eliminating redundant actions.

Baram, Tennenholtz, and Mannor [5] explored redundancy avoidance in reinforcement learning through transition entropy and tried to eliminate the redundancies in both deterministic and stochastic settings through MDPs, an actor-critic framework, and Q-Learning. Actions are given an action redundancy score (deterministic) and an action redundancy ratio (stochastic). The actor-critic framework attempts to update the action redundancy score and action redundancy ratio using a transition buffer but uses an old policy. The proposed Q-Learning algorithm was created to fix that issue.

The work of Zahavy [6] takes a different approach by using deep neural networks to eliminate redundant actions. A binary signal is created based on auxiliary rewards through a Markov decision process that determines if an action should be chosen.

The gaming field is another area where redundant action avoidance is being researched. The primary technique used is Monte-Carlo Tree Search. Santos, Bernardino, and Hauck [7] sought to improve the Rolling Horizon Algorithm with a shift buffer to redistribute actions by shifting actions and generating new actions. The redundancy avoidance is applied to the Monte-Carlo Tree Search, which uses the actions from the shift to find the best possible next actions by testing all actions at the current state.

The work explored in [8] uses the Monte-Carlo Tree Search to focus not only on redundant action avoidance but also on loss avoidance. This process involves pruning the tree such that a minimum amount of actions and the associated losses are kept. The pruning also accounts for nodes generated by Monte-Carlo Tree Search that are deemed redundant.

III. Active Logic

Active logic [9] differentiates itself by being an agent's internal reasoner [10] with the ability to keep track of the passage of time and diffuse direct contradictions [11]. The core of active logic is first-order logic with inference rules to maintain the evolving Knowledge Base (KB) current. The facts in active logic are not set in stone, and every fact that is stored within the agent's KB can be accessed and revised at a later time.

Active logic maintains the passage of time using the clock rule that keeps track of the current time. The clock rule, $now(t) \rightarrow now(t+1)$ uses the fact that now the time is t at time t, to infer that now has become t+1 at time step (t+1). At time t, the agent has a record of the current beliefs and observations within the agent's KB. Under normal circumstances, these get inherited to the next time step. Additionally, using Modus Ponens on the existing knowledge at time t, new facts are inferred and asserted into the KB at time t+1. Thus, at each time step, an agent will have a record of when each piece of knowledge was gained.

Active logic also has a distinct feature of contradiction detection and handling. Agents cannot have two conflicting beliefs at the same time. Active logic distrusts conflicting beliefs and prevents new knowledge from being derived from them until the conflict is resolved. Suppose an active logic sentence P exists in the KB when a new inference (or observation) $\tilde{}P$ occurs at time t. This causes conflict in the agent's beliefs, and the logic distrusts both P and $\tilde{}P$ and asserts $contra(P,\tilde{}P)$ in its KB at time step t+1. The predicate, $contra(P,\tilde{}P)$, serves as a way to note in the KB that a direct contradiction between P and $\tilde{}P$ has occurred.

IV. Action-Selection Axiom sets

An active-logic based agent goes through a $perceive - think - act$ cycle to perform its tasks. The cycle repeats itself, allowing the agent to keep executing actions. The subset of axioms that the agent uses for action selection within the $think$ phase is the focus of this paper. This section discusses six sets of action-selection axioms [12], along with how each axiom set handles repeated action avoidance. The first is the baseline action selection, which contains no knowledge for avoiding repeated actions. This is followed by the remaining five axiom sets that contain knowledge for avoiding repeated actions. When improving the baseline action selection, the core pattern was preserved and adjusted to increase the amount of caution towards repeated actions within the agent. Each pattern includes an additional piece of knowledge that affects the agent's behavior leading to potentially more stringent repeated action avoidance. Repeated action avoidance is handled using five different action-selection axiom sets with increasing levels of caution. The different Repetition Avoidance (RA) axiom sets are RA 2, RA 2-3, RA 2-4, RA 3-4, and RA 3-5.

Each axiom set has a corresponding sequential block diagram that outlines the conditions needed to select an action. The diagrams flow downward through vertical arrows from one level to the next. Each level within the diagram represents a time step during reasoning and is separated by dotted lines. On the left of each level is its corresponding time step, $T+1$, $T+2$, etc. The diamonds in the figures represent knowledge checks that occur at the time step, while the blue rectangles symbolize knowledge being asserted at the time step, and a green rectangle indicates beliefs inherited from the previous steps. Multiple arrows from an entity indicate an and condition.

Activities associated with every block that appears at the same time step occur simultaneously. For example, both precondition and post-condition checks happen independently but simultaneously at T. At the first time step of reasoning, some knowledge checks will automatically fail as that knowledge has not been asserted yet. However, as time passes and new observations come in, when the same rules fire at a later time step, the knowledge conditions (the antecedents) may hold true for the consequent to get asserted in the KB.

An action that can be executed by the agent is noted in the KB as a *can do action*. Once an action has been selected amongst the *can do actions*, the action-selection specific inference knowledge is removed, and the entire process of selecting an action automatically activates again with a new set of *can do actions*. The action-selection specific inference knowledge in the figures that are removed when an action is selected include *can do actions*, *feasible actions*, *original actions*, *duplicate actions*, and *contingent actions*.

A. Baseline - No Repetition Avoidance (Naive 2 Steps)

The baseline action-selection axiom set infers action whose preconditions are met as possible *can do actions*. For the sake of this paper, this axiom set will be called Naive. Naive takes a total of two time steps for the reasoning to select an action. The only requirement to start reasoning about an action is having current beliefs. When the agent has a new belief, the agent will check whether both post-conditions and preconditions are satisfied separately. An example of a precondition for a move action is the availability of an empty location to move to. When the preconditions are met, a possible action is now available to be performed. If the post-conditions of an action for an object are met, then the action is marked as completed. An example of the post-condition of a move action to a location is for the agent's current position to change to that location.

Naive is the most extreme of the bold action-selection axiom sets, focusing on selecting an action whose preconditions are met. If there are multiple actions with preconditions met available at any time step, one will be selected at random. Due to the naive nature of the baseline action-selection axiom set, the agent will act randomly within the environment. If the neighboring location being considered is empty, there is the possibility that the agent moves to that location with no regard to whether or not the action is a repeated action. As a result, the agent can repeatedly search areas where it has been searching, causing the agent to potentially fail the task. In extreme cases, it is possible for the agent to move in a circular motion and waste precious time because it has no restrictions on where to move.

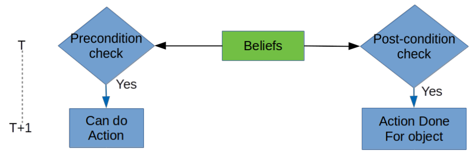


Fig. 1. Naive reasoning abstraction

B. Only Non-duplicated Actions (2 Steps Deliberation)

The 2 Steps Deliberation for action selection, formally named RA 2, completely eliminates any repeated actions by building upon the structure of the baseline pattern and introducing an extra-knowledge check. While the baseline action-selection axiom set selects any action whose preconditions are met, RA 2 selects only those actions that are not repeated as possible *can do actions*. At every time step, the agent performs a check to determine if the post-conditions of an initiated action are met, and when the check succeeds, it stores that knowledge. This knowledge is inherited in future time steps, and hence if an action was executed in a previous $perceive-think-act$ cycle, that knowledge is accessible for later use. The agent uses this knowledge in the subsequent $perceive-think-act$ cycles to check if an action was performed for an object; if the check is true, the agent will not redo that action for the object. RA 2 does not increase the number of time steps in the $think$ phase for action selection when compared to the naive method. However, the agent becomes more cautious due to repetition avoidance.

In the search task example, this axiom set can cause the agent to become boxed in. The agent can potentially visit surrounding areas in one direction, which forces the agent to move to one side of the environment. When all its immediate neighboring locations are visited, the agent has no new locations to move into without stepping on an already visited location. Hence, it becomes stuck at its position and is unable to complete its task frequently.

C. Non-duplicated actions inferred one step before other feasible (preconditions met) actions (2-3 Steps Deliberation)

While the 2-3 Steps Deliberation method (also called RA 2-3) uses the core structure from the previously introduced action-selection axiom sets to select those actions that are not repeated as possible *can do actions* in two steps, it takes a different path by asserting additional knowledge for

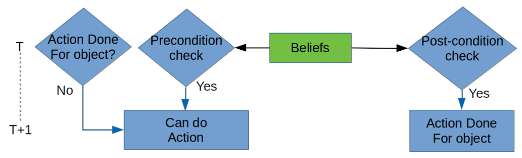


Fig. 2. RA 2 reasoning abstraction

each action whose preconditions are met as a *feasible* action. This knowledge is used to select actions to repeat when no action that has not been done previously is available. Thus, a *feasible action* becomes a possible *can do action* in a $perceive - think - act$ cycle as long as no other action got selected within the $think$ phase in two steps. If the agent infers *can do actions* in two steps, then the $think$ phase ends, and the agent transitions to the act state. With this adjustment, the agent can perform feasible actions that the agent had previously done for an object when no actions that were not done previously exist for that object. This differentiates it from RA 2, which becomes inoperable in the same situation. Since a possible repeated action is derived one time step after an action is marked as feasible, while actions that are not attempted previously are immediately selected for execution, the original actions will be preferred over repeated actions by agents using this axiom set. The combination of these concepts requires at least two or three time steps of action-selection reasoning in the $think$ state.

Using the search task example, RA 2-3 uses a relatively balanced approach but leans towards being bold as it attempts to prioritize speedy action. An empty location that has not been visited before, if available, is immediately chosen as a possible location to move to. RA 2-3 takes a different approach to repeated action avoidance by utilizing the stored *feasible actions* knowledge as a means of selecting previously tried actions. For the target search task, the agent notes empty neighbors as feasible locations to move to. This knowledge is used to infer possible locations to move to if the $think$ phase doesn't provide a possible action that has not been done before, i.e., an empty neighboring unvisited location to visit.

D. Non-duplicated actions preferred over feasible duplicated actions (2-4 Steps Deliberation)

The 2-4 Steps Deliberation method (RA 2-4) is very similar to the 2-3 deliberation method because they both add new means of performing an action when an action was previously done for an object. The core structure from the previously introduced action-selection axiom sets remains, but *duplicate actions* are asserted instead of immediately selecting a repeated action. In RA 2-4, *duplicate action* knowledge is derived at the same time step that repeated actions are selected in RA 2-3. The selection of repeated actions from *duplicate actions* becomes possible one time step later. This leads to the $think$ state reasoning for action selection taking at least two or four time steps instead of two or three. A *duplicate action* will only be executed if it is also a *feasible action*.

By asserting *duplicate actions*, the action selection process gains a new level of caution towards repeated actions because the agent will have more time to think about what is occurring. RA 2-4 will still prioritize actions not previously done before, similar to how RA 2-3 performs. RA 2-4 shares the same problem of making decisions too quickly while considering actions that have not been previously selected, as in RA 2-3. The difference is the agent has more time to think about selecting a repeated action due to noting that the action is a *duplicate action*, but RA 2-4 still performs new actions potentially too quickly. When an action was not previously done, a possible action is asserted immediately, which is given the highest priority. *Duplicate actions* will primarily play a role when the agent becomes stuck or when a new action is currently not available. While the RA 2-3 action-selection axiom set prioritizes speed, the RA 2-4 action-selection axiom set prioritizes slightly more knowledge representation and reasoning for repeated actions. With this being a middle ground for the action-selection axiom sets, it is the perfect balance of boldness and caution for agents.

E. Feasible actions marked as original or duplicate; original preferred over duplicate (3-4 Steps Deliberation)

The 3-4 deliberation method (RA 3-4) appears the most different from the previous action-selection axiom sets, but the concept from RA 2-4 remains in this axiom set. The important addition is asserting additional information on the outcomes of checking if an action was done for an object. This is accomplished through *duplicate actions* similar to RA 2-4 but also asserting *original actions* when an action is not previously done for an object. Thus, the possible action from an action not previously done is also pushed down a time step to accommodate for checking if the predicate *original action* exists in the knowledge base at the time. An *original action* will always take precedence over a *duplicate action* if both are available. The *original actions* can also only be executed if they are also *feasible actions*.

Due to the addition of asserting *original actions*, RA 3-4 becomes much more cautious as three time steps are needed to

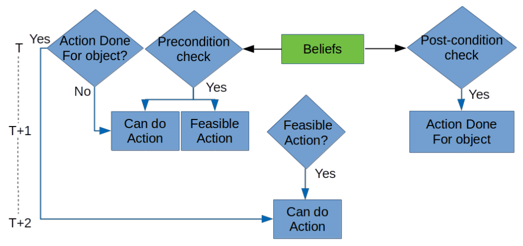


Fig. 3. RA 2-3 reasoning abstraction

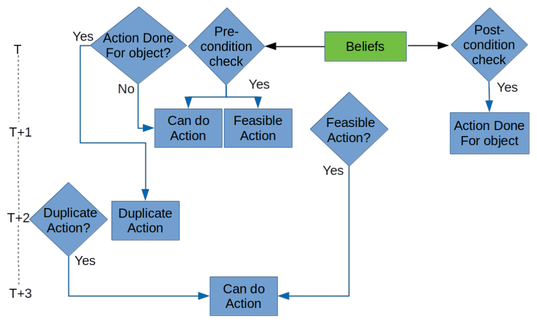


Fig. 4. RA 2-4 reasoning abstraction

select an action even in the best case. The main characteristic of this axiom set is the alternative option for correcting the boxed-in scenario present within RA 2 and addressing the shortcomings of RA 2-3 and RA 2-4 by introducing *original actions* and *duplicate actions* for the agent to consider. In every situation, an *original action* is prioritized over a *duplicate action* because the agent is less likely to complete the task by revisiting locations. If there are only inspected locations surrounding the agent, the agent must do the *duplicate action* to find possible overlooked areas. One possible downside to this action-selection deliberation is an *original action* arriving late into the KB (perhaps preconditions became true a few steps into the *think* phase), forcing the agent to select the *duplicate action*, resulting in possibly a sub-optimal move. While this is not an inherent flaw of the reasoner, it is worth mentioning as the agent does not know that waiting longer will result in an *original action*.

F. Feasible actions marked as original or duplicate; feasible duplicates marked as contingent; original preferred over contingent (3-5 Steps Deliberation)

The core of the 3-5 deliberation method (RA 3-5) is almost exactly the same as RA 3-4. This action-selection axiom set makes the agent represent and reason with extra knowledge for selecting a repeated action. Instead of going immediately to a possible *can do action* from a *duplicate action*, it asserts a *contingent action* in the next time step. The *contingent action* will only result in a possible action if no *original actions* have appeared and the action is also a *feasible action*. Once there is a *duplicate action* present, then the *contingent action* knowledge is derived in the KB. Like the 3-4 deliberation step-logic, an *original action* still takes precedence over other actions. In the best case scenario, the *think* state can produce a possible action in three time steps when actions that have not been attempted are available to select from. When such actions are not present, the agent will take at least five time steps to select a repeated action.

The 3-5 Step reasoning uses all of the previous patterns and addresses the issues with each action-selection axiom set. By using both *duplicate action* and *original action* after an action is done, the issue of being too bold in some cases is solved. To address the issue of the agent needing more time to process *duplicate actions*, *contingent actions* were used to make the agent wait one extra time step before asserting a possible action through *duplicate actions*. The result of the combination of knowledge is extreme caution, which is reflected in the amount of time needed to perform any action.

With this being the most cautious of the action-selection axiom sets, the agent takes extra time steps to select the action to return to a previous location. In the search task, this allows going to unexplored locations that lay beyond the visited ones only when immediate neighbors need not be explored. The extra steps for repeated actions in the *think* step allows more time for inferring new actions that could be selected. There is a chance that waiting more time steps could also lead to similar results, but this can lead to a slippery slope. The addition of more time steps could stall the agent enough that it rarely completes its task.

V. Experiment

This experiment uses an active-logic based agent tasked with finding a unique object within a virtual environment (AI2-THOR) using one of the six action-selection axiom sets. Agents have a deadline of 100 time steps to locate the target during a trial. There are a total of 20 trials containing different starting locations for either the agent or the target. These 20 trials are also repeated as a set, five times for each pace length and axiom set. Pace length represents the units needed for an agent to move to a new location in the environment. For this experiment, the pace lengths used were .2 through .45, with an interval of .5 pace length. Effectively, there were six different pace lengths used. Each set was also repeated three times during the experiment. Thus, every action-selection axiom set was used to perform a total of 1,800 trials.

VI. Results

The results from Fig. 7 contain the average success of finding the target for each action-selection axiom set. The higher numbers (brighter colors) represent better success in finding the target. A successful trial requires an agent to find the target and infer that the task has been completed. If the target has been found within the deadline, but the completion

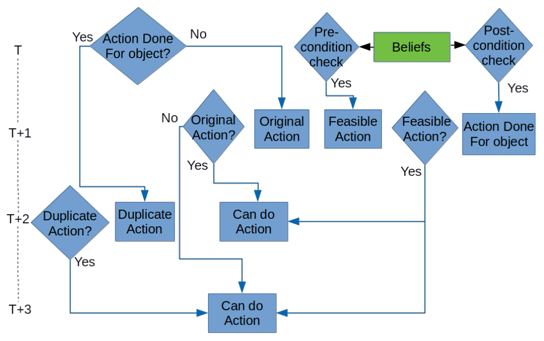


Fig. 5. RA 3-4 reasoning abstraction

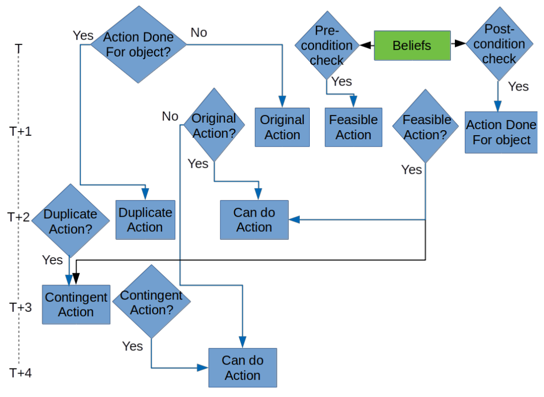


Fig. 6. RA 3-5 reasoning abstraction

of the task has not been inferred during the deadline, the trial will result in failure. The extremes of bold and cautious axiom sets did not perform as well as the middle ground axiom sets. The bold axiom sets struggled to find the target due to acting without any knowledge or failing to find actions. More specifically, RA 2 performed similarly to the balanced axiom sets at lower pace lengths but started to perform much worse at the higher pace lengths. At the lower pace lengths, it is much harder to become boxed in because of the amount of movement required to cover the environment. The opposite becomes true at higher pace lengths as coverage of the environment will take much less time, creating more opportunities to become boxed in. Overall, increasing the pace length had a positive impact on the performance of all axiom sets.

Fig. 8 shows how quickly an agent using one of the action-selection axiom sets completed the task. Unlike Fig. 7, the higher numbers, in this case, represent poor performance. This set of results does not include any trial where the agent fails to find the target. Thus, the times shown will not include any time that an agent has failed to meet the deadline, resulting in reduced times. This is important because RA 2 has the best speed for finding the target but fails to find the target often. With that in mind, RA 2 does complete the task very quickly when it does find the target. When using both Fig. 7 and Fig. 8, the balanced axiom sets produced the best results overall. The completion times of the cautious axiom sets were hindered by the extra reasoning time, which was the expected outcome. The extra reasoning time likely also played a factor in failing to complete the task.

The heatmap in Fig. 9 highlights the average percentage of new actions performed within the environment during the trials. By design, RA 2 eliminated all repeated actions as possible actions as it was deemed futile. Thus, RA 2 selected a new action whenever possible, resulting in a new action percentage of 100%. Also, as expected, Naive performed new actions close to 50% of the time. If all actions have equal weight of selection, the deciding factor is only what is present at a given time step. Therefore, there will be time steps where an agent will only have new actions or only repeated actions. The remaining axiom sets performed actions at a progressive rate overall, selecting slightly more new actions than the

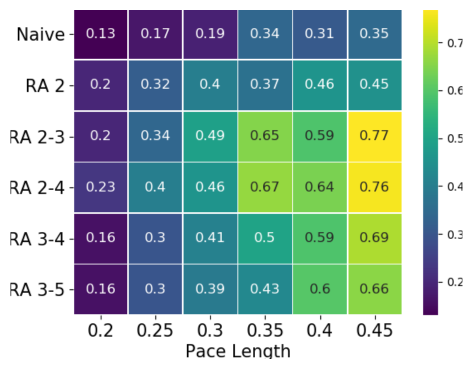


Fig. 7. Accuracy of finding the target

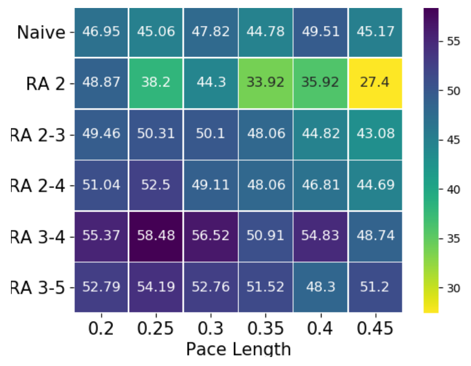


Fig. 8. Average time steps to find the target

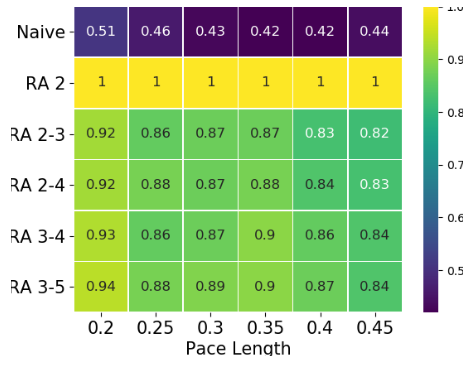


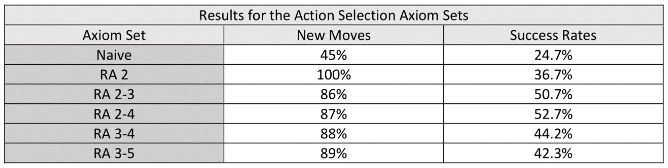
Fig. 9. Percentage of original actions selected

next. Fig. 10 contains the average percentages of new action performed during the experiment.

Overall, the balanced axiom sets performed the best at using the knowledge to avoid repeated actions while completing the task effectively. The best-performing axiom set was RA 2-4, while Naive resulted in the worse performance in every category. RA 2 actually hinders its performance by eliminating all repeated actions. The elimination of repeated actions prevents returning to locations to find undiscovered areas or causes additional time to avoid the action. The additional time added is only a problem if the agent is unable to find the target due to the process. This is also true when referring to avoiding futile actions. Increasing the time, whether it is selecting actions to avoid the futile action or spending additional time to think about the best option, will hurt the potential speed of finding the target; but it will only potentially harm the success rate. Therefore, when comparing the results for the cautious and balanced axiom sets, the cautious axiom sets had a relatively close performance. Fig. 10 shows the average success rate for each axiom set. Based on the success rate, the additional knowledge added in RA 3-5 starts to hurt the performance. Thus, increasing the time to reason about repeated actions will increasingly make the agent fail its task more.

VII. Conclusion

We discussed several axiom sets for action selection that represent and reason with varying knowledge content for avoiding futile repeated actions. We examined how an agent



Results for the Action Selection Axiom Sets		
Axiom Set	New Moves	Success Rates
Naive	45%	24.7%
RA 2	100%	36.7%
RA 2-3	86%	50.7%
RA 2-4	87%	52.7%
RA 3-4	88%	44.2%
RA 3-5	89%	42.3%

Fig. 10. Average new moves and success rate percentages

using these axiom sets will have its knowledge evolve as time progresses and how the evolving knowledge affects task performance using a simple target search task. This research examined the strength and weaknesses of each axiom set and discussed the performance of time-situated agents using these axiom sets in a search task. Further work to analyze the behavior of multiple active-logic based agents collaborating in the environment on the same task is ongoing. In the future, we will study the interactions of more complex knowledge (axioms, inferences, and observations) with task performance and success rates for active-logic based agents.

ACKNOWLEDGMENT

This work was supported by DARPA Project No. HR00112090025, U.S. Department of Education Title III HBGI Grant, and ArtIAMAS Subaward No. 11788-Z8523201, Prime Award No. W911NF2120076. Special thanks to D. Perlis, M. Goldberg, C. Maxey, T. Clausner, J. Conroy, D. Conover, N. Vale, and J. Milzman for their input.

REFERENCES

[1] M. Nirkhe, S. Kraus, and D. Perlis., “Thinking takes time: A modal active-logic for reasoning in time,” in *Proceedings of BISFAI-95*. AAAI Press, 1995.

[2] D. P. Josyula, A. Herron, and K. M’Bale, “Implementing a task-oriented time-situated agent,” in *Eighth Annual Conference on Advances in Cognitive Systems*, 2020.

[3] L. Chrpa, T. L. McCluskey, and H. Osborne, “Determining redundant actions in sequential plans,” in *2012 IEEE 24th International Conference on Tools with Artificial Intelligence*, vol. 1. IEEE, 2012, pp. 484–491.

[4] T. Balyo, L. Chrpa, and A. Kilani, “On different strategies for eliminating redundant actions from plans,” in *Seventh Annual Symposium on Combinatorial Search*, vol. 5, 2014, pp. 10–18.

[5] N. Baram, G. Tennenholtz, and S. Mannor, “Action redundancy in reinforcement learning,” in *Uncertainty in Artificial Intelligence*. PMLR, 2021, pp. 376–385.

[6] T. Zahavy, M. Haroush, N. Merlis, D. J. Mankowitz, and S. Mannor, “Learn what not to learn: Action elimination with deep reinforcement learning,” *Advances in neural information processing systems*, vol. 31, pp. 3566–3577, 2018.

[7] B. Santos, H. Bernardino, and E. Hauck, “An improved rolling horizon evolution algorithm with shift buffer for general game playing,” in *2018 17th Brazilian Symposium on Computer Games and Digital Entertainment (SBGames)*. IEEE, 2018, pp. 31–316.

[8] D. J. Soemers, C. F. Sironi, T. Schuster, and M. H. Winands, “Enhancements for real-time monte-carlo tree search in general video game playing,” in *2016 IEEE Conference on Computational Intelligence and Games (CIG)*. IEEE, 2016, pp. 1–8.

[9] D. Perlis, K. Purang, D. Purushothaman, C. Andersen, and D. Traum, “Modeling time and meta-reasoning in dialogue via active logic,” in *Working notes of AAAI Fall Symposium on Psychological Models of Communication*, 1999.

[10] D. Perlis, J. Brody, S. Kraus, and M. Miller, “The internal reasoning of robots,” in *Thirteenth International Symposium on Commonsense Reasoning*, 2017.

[11] J. J. Elgot-Drapkin and D. Perlis, “Reasoning situated in time I: basic concepts,” *Journal of Experimental Theoretical Artificial Intelligence*, vol. 2, pp. 75–98, 1990.

[12] A. Herron and D. P. Josyula, “An analysis of the deliberation and task performance of an active logic based agent (student abstract),” in *Thirty-Seventh AAAI Conference on Artificial Intelligence*. AAAI, 2023.