S4H: A Tool for Synthesizing Human-Like Strategies

Marco Aruta[1], Vadim Malvone[2], and Aniello Murano[1]

[1] Università degli Studi di Napoli Federico II, Naples, Italy
[2] LTCI, Télécom Paris, Institut Polytechnique de Paris

Abstract. In the era of Human-like AI, creating intelligent systems that mimic natural human decision-making has become a pivotal research goal. A key challenge is the design of tools that align with human cognitive limitations to foster smoother interactions and effective collaboration in human-centered environments. In this paper, we present *Strategies for Humans* (*S4H*), the first formal-methods tool for synthesizing *human-like* rational strategies, leveraging *Natural Alternating-time Temporal Logic* (*NatATL*), a logic recently introduced to reason about bounded strategies. By designing and implementing novel strategies generation techniques tailored to human cognitive constraints, our tool bridges the gap between theoretical formalism and practical AI applications. To demonstrate its effectiveness, we conduct experiments on real-world models, showcasing its ability to synthesize human-like strategies efficiently. Our contributions mark a step forward in the development of socially-aware, sustainable AI systems, particularly in domains like social robotics and human-in-the-loop applications.

Keywords: Human-like AI · Natural Strategies · Strategy Logic.

1 INTRODUCTION

The rapid development of human-like AI has emerged as a crucial research direction, aiming to create intelligent systems that can reason and act in ways closely aligned with human cognitive processes. This approach promises smoother human-machine interactions and fosters collaboration in environments where human decision-making is inherently bounded. By intentionally limiting the computational power and strategy complexity of artificial agents, human-like AI ensures that decisions remain intuitive and relatable, thereby contributing to a sustainable technological ecosystem.

A recent innovation in this field is *Natural Alternating-time Temporal Logic* (*NatATL*) [14], which extends the traditional ATL framework by introducing bounded strategies via the operator $\langle\langle A \rangle\rangle^{\leq k}\phi$, where $k \in \mathbb{N}$ denotes a complexity bound. This bounded strategy concept is designed to mirror the intrinsic bounded rationality observed in human decision-making. Beyond serving as a formal verification framework for multi-agent systems, *NatATL* opens a novel perspective on evaluating human-likeness: by empirically establishing an upper

limit on strategy complexity, it becomes possible to differentiate between human and artificial agents. This approach even offers a new angle on the Turing Test, suggesting that the complexity of an agent's reasoning process could be a criterion for assessing its human-like behavior.

Our Contribution. In this paper, we introduce *Strategies for Humans* (*S4H*), a novel formal-methods tool, leveraging on NatATL, for synthesizing human-like strategies. S4H represents a first standard formal approach for bounded strategic reasoning in practice. Through the dynamic generation of strategies inspired by human cognitive behavior, our tool aims to contribute to bridging the gap between theoretical bounded-rationality models and their practical use in AI systems. At the core of our tool, we have developed three key components:

- **Dynamic Strategy Generation:** An algorithm that incrementally synthesizes bounded strategies, starting from the simplest and gradually considering more complex ones only when needed. This mirrors the human tendency to prefer simpler decision processes.
- **Model Pruning:** A pruning technique projects synthesized strategies onto the system model, reducing its complexity while preserving correctness. This step ensures that only feasible, strategy-compliant transitions are considered during verification.
- ***ATL* Pre-processing Optimization:** By integrating a traditional *ATL* verification phase as a pre-processing step, our approach filters out unsatisfiable instances early. This dramatically improves the efficiency of the *NatATL* verification process.

Our tool is especially suited for applications in social robotics, multi-agent distributed systems, and human-AI collaboration scenarios-contexts where decision-making under bounded rationality is critical. To validate our approach, we applied it to real-world case scenarios, including an extensive case study on a logistics robot system. The experimental results will discuss not only the scalability and practical applicability of our tool but also the significant performance gains obtained from our proposed optimization techniques.

Related Works. Model checking has played a crucial role in strategic reasoning within Multi-Agent Systems (*MAS*), with tools like *NuSMV* [1], *SPIN* [11], and *UPPAAL* [8] providing verification frameworks for finite-state, distributed, and real-time systems. *MOCHA* [24] and *MCMAS* [19] have further extended verification to strategic interactions using *ATL*, with enhancements supporting epistemic and temporal epistemic logics [16, 20, 22]. However, these tools do not address bounded rationality, a key aspect of human-like decision-making. Bounded rationality, introduced by Simon [3], accounts for cognitive and computational constraints in decision-making. Traditional AI models assume unbounded rationality, whereas real-world decision-making is constrained by time and resources [9,10]. Recent and promising tools like VITAMIN [7] introduced verification capabilities adaptable to various logics. Works on heuristic-based AI [17,27],

satisficing models [6, 26], and cognitive architectures [18, 23] has emphasized the importance of limiting computational complexity to achieve human-like behavior. The role of bounded rationality has also been explored in distinguishing human and artificial agents in the context of the Turing Test [5]. Applications extend to social robotics and human-in-the-loop AI, where agents must operate under cognitive constraints to ensure intuitive and predictable behavior [2, 4]. Our tool integrates verification capabilities for *NatATL*, bridging the gap between model checking and human-like AI. By addressing bounded rationality, we provide a novel tool for analyzing strategic interactions with computational feasibility and interpretability, strengthening the foundation for future research in explainable AI.

Outline. The rest of the paper is organized as follows. Section 2 recalls the main notions of natural strategies and *NatATL*. Section 3 presents the implementation of our tool. Section 4 evaluates the performance of our module through experiments. Finally, we conclude in Section 5.

2 PRELIMINARIES

In this section, we provide the necessary background to understand the proposed tool introducing *Natural Alternating-Time Temporal Logic* (*NatATL*). We include detailed definitions and explanations to clarify key concepts and ensure the self-containment of this paper. Readers interested in additional formal details can refer to [13–15].

We start with the definition of our models.

Definition 1. *A Concurrent Game Structure (CGS) is a formalism used to model Multi-Agent Systems (MAS), where agents make decisions simultaneously. Formally, a CGS is defined as a tuple:*

$$S = \langle Agt, Q, \Pi, \pi, d, \delta \rangle$$

where:

- ***Agt**: A finite set of agents, with $|Agt| = n \geq 1$.*
- ***Q**: A finite set of states.*
- *Π: A finite set of atomic propositions.*
- *π: A labeling function $\pi : Q \to 2^{\Pi}$, assigning to each state the set of propositions true in that state.*
- ***d**: A function $d : Q \times Agt \to \mathbb{N}$, indicating the number of available actions for each agent in each state.*
- *δ: A transition function $\delta : Q \times (\prod_{a \in Agt} d_a) \to Q$, mapping a state and a vector of actions to a successor state.*

Given our model, we define natural strategies and their complexities.

Definition 2. *A natural strategy for an agent $a \in Agt$ is a rule-based system that dictates actions based on conditions. Formally:*

- ***Memoryless Strategies (nr-strategies):*** *These rely only on the current state and are defined using Boolean formulas over atomic propositions.*
- ***Strategies with Recall (nR-strategies):*** *These consider the history of states, using regular expressions over atomic propositions to define conditions.*

The complexity of a natural strategy is determined by the size of its representation:

- *For nr-strategies: The total number of symbols in the Boolean conditions.*
- *For nR-strategies: The total size of all regular expressions used.*

From now on, in this paper, we will refer to each strategy type using the abbreviations nr and nR. Given the notion of natural strategy, we can provide a logic that uses these strategies.

Definition 3. *NatATL extends Alternating-Time Temporal Logic (ATL, [25]) by introducing a bounded rationality operator $\langle\langle A\rangle\rangle^{\leq k}\phi$. This operator asserts that a coalition of agents $A \subseteq Agt$ can enforce ϕ using strategies with complexity at most k. The grammar of NatATL is given by:*

$$\phi ::= p \mid \neg\phi \mid \phi \wedge \phi \mid \langle\langle A\rangle\rangle^{\leq k} X\phi \mid \langle\langle A\rangle\rangle^{\leq k}(\phi U \phi)$$

where:

- *$p \in \Pi$ is an atomic proposition.*
- *X and U are temporal operators meaning "next" and "until," respectively.*
- *$k \in \mathbb{N}$ is the complexity bound for the strategies of agents in A.*

We can conclude this section with *NatATL* semantics.

Definition 4. *A state $q \in Q$ satisfies a NatATL formula ϕ, denoted as $S, q \models \phi$, if ϕ holds starting from q under the strategies defined for the coalition. The satisfaction relation is defined inductively:*

- $S, q \models p \iff p \in \pi(q)$.
- $S, q \models \neg\phi \iff S, q \not\models \phi$.
- $S, q \models \phi_1 \wedge \phi_2 \iff S, q \models \phi_1$ *and* $S, q \models \phi_2$.
- $S, q \models \langle\langle A\rangle\rangle_{\leq k} X\phi \iff \exists s_A$: *a collective strategy for A with complexity compl* $\leq k, \forall q' : (q \xrightarrow{s_A} q') \implies S, q' \models \phi$.

3 IMPLEMENTATION

In this section, we present the algorithms that form the core of the proposed *S4H* tool. The implementation is designed to be consistent with the theoretical foundations established in [14], ensuring that the practical aspects align with the formal semantics of *NatATL*. As we mentioned previously, our tool is structured around three main Python-based procedures: Strategies Generation,

Model Pruning, and Model Checking. The process begins by systematically evaluating each possible strategy. For each collective strategy s_A, the input model is refined by eliminating transitions that do not conform to s_A. Subsequently, model checking algorithmsare applied, leveraging their polynomial complexity to achieve efficient verification. A high-level overview of this workflow is depicted in Fig. 1.

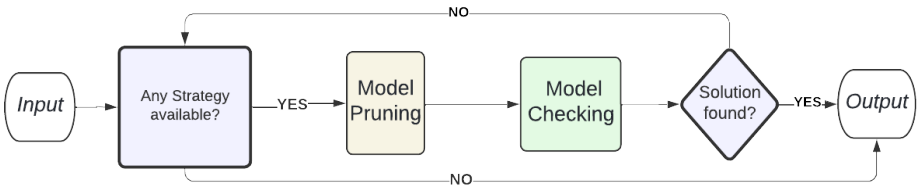

Fig. 1. *NatATL* Model Checking process

3.1 Strategies Generation

The generation of natural strategies is a crucial component of our approach, directly addressing the objective of modeling *human-like behavior*. Each strategy produced is guaranteed to respect the bounded complexity constraint k, ensuring soundness while maintaining computational feasibility. As defined in [14], natural strategies are mappings from states to actions, determined by Boolean conditions (for *nr-strategies*) or regular expressions (for *nR-strategies*). Our algorithm brings these theoretical foundations to life by systematically constructing condition-action pairs within the given complexity bound.

A fundamental challenge in this process is the vast space of possible strategies and their varying complexities. Since our goal is to approximate human decision-making behavior, we must consider a realistic subset of strategies that a coalition of agents might adopt. To optimize this process, we introduce a dynamic strategy generation function. This function incrementally generates strategies, starting with those of minimal complexity (that is, the lowest value of k) and only expanding to higher complexity levels when no valid solutions are found in simpler instances. This mirrors a bounded rationality approach, reflecting how humans prioritize simpler strategies before considering more complex ones. In addition, this dynamic and incremental approach provides two key advantages: *computational efficiency* and *optimization*. By prioritizing lower k values, our method avoids exhaustive enumeration of all possible strategies up to a fixed upper bound, significantly reducing the number of generated strategies. This segmentation facilitates the identification of the smallest winning strategy, improving both efficiency and interpretability in the verification process. The next section details how a pruning process enhances the verification step by reducing the state space while preserving logical correctness.

3.2 Model Pruning

This task involves projecting each collective strategy onto the model. The model is pruned according to the actions chosen by the agents in the coalition A, removing all moves incompatible with the strategies examined. To clarify this phase, we distinguish between two approaches: pruning for nr-strategies and pruning for nR-strategies.

nr-Strategies Model Pruning. For each selected s_A, we project the strategies only onto the model states where the current strategy conditions are met. To ensure that each agent always has an action to execute, even when none of the conditions of its current strategy is satisfied, we include an additional condition-action pair, $(\top, idle)$. This pair allows the agent to perform the idle action by default in any state, relating to its true condition (i.e., applicable to all states) whenever no other action can be performed. If a current strategy includes an action that is not present in the current model states (where its conditions are met), the collective strategy is discarded as invalid. Upon completion of this iterative process, the algorithm returns the pruned model. Any checks for the validity of the strategy are performed by external functions.

nR-Strategies Model Pruning. Unlike the theoretical approach that involves the use of Automata and a different method for solving model checking for *NatATL* with recall (check [14] for more details), in this work, we propose a different approach that involves using tree structures instead of automata. This approach does not undermine the method shown with nr-strategies but rather leverages the potential of this representation to capture the concept of history. From a practical and algorithmic standpoint, using a tree can be preferred over an automaton for several reasons. Firstly, tree structures can naturally represent hierarchical data and branching processes, which aligns well with the concept of history and state transitions in many applications. Secondly, trees can provide more efficient traversal and manipulation algorithms, allowing faster updates and searches. Lastly, trees facilitate a more intuitive visualization and understanding of the state space, making it easier to debug and optimize the model. So, for the model pruning approach in this context, we unrolled the initial *CGS* into a tree structure to better represent the concept of histories and ensure that we keep track of all transitions and the different paths that arise from the base model, thus taking repetitions into account. After model conversion, we project the current collective strategy onto the tree. Each individual strategy within a collective strategy must be validated before pruning. Strategies with actions not present in the model for states where the condition is met are considered invalid. Subsequently, each individual strategy prunes the tree, potentially reducing its size for the next iteration and the next agent's strategy. A valid collective strategy may become invalid for a reduced model, requiring a correctness check each time an agent operates on an individual strategy. Pruning depends on the current guarded action's condition, whether a boolean formula or a regular expression. For regular expressions, pruning occurs along the nodes satisfying the

expression's path. If the condition is a boolean formula, all subtrees from paths inconsistent with the selected action are removed for nodes whose state matches the condition. To ensure that an agent can act each turn, if no guarded action is applicable within a strategy, a supplementary condition-action pair ($\top$, *idle*) is added, allowing each agent to perform *idle* in every state. This ensures at least one possible action per turn. The initial *CGS* construction ensures each agent can perform *idle* in every state. The pruned tree is reconverted into an extended *CGS*, with unique state renaming to avoid repetition, resulting in the desired *CGS* for model checking.

3.3 Model Checking

The model checking algorithm takes as input a specified formula and the refined model obtained from the previous pruning step. The implemented algorithm is based on the theoretical foundations presented in [12], which provide further details. Our approach employs a tree structure to represent the parsed formula, utilizing a depth-first search strategy to traverse the tree and a bottom-up evaluation to process logical formulas with temporal operators. Each node in the tree corresponds to a subformula, and the algorithm computes the set of states that satisfy it.

A key feature of our approach is that, beyond returning a Boolean answer indicating whether a solution exists for the given model, the algorithm also synthesizes the optimal winning strategy leading to that solution. Specifically, if a solution validating the formula is found, the tool returns both the set of states that satisfy the root formula and the minimal-complexity strategy that guarantees its satisfaction. This process ensures that, rather than merely verifying the existence of a strategy, *S4H* actively constructs and returns an executable strategy that can be followed by agents in the system. If the verification does not yield a solution for the pruned model, the algorithm backtracks to the Strategies Generation phase, projects new strategies onto the original model, and reattempts the verification. This iterative cycle continues until a valid strategy is synthesized or no further strategies remain viable. For the complete implementation, please visit the project directory at the following link: https://github.com/MarcoAruta/S4Htool.

3.4 Pre-processing Optimization

S4H tool introduces an optimized verification approach tailored to domains where modeling bounded rationality and strategic decision-making is crucial, such as automated planning, human-AI collaboration, and strategic interactions in multi-agent systems (*MAS*). By integrating Alternating-Time Temporal Logic (*ATL*) and Natural Alternating-Time Temporal Logic (*NatATL*) within a preprocessing verification, we achieve a balance between computational efficiency and expressive power. This optimization is particularly significant for real-world applications requiring scalable and interpretable strategy verification. The key contributions of this approach include:

- *Efficient pre-processing via ATL*: ATL serves as an initial filtering mechanism to rapidly discard unsatisfiable paths, significantly reducing the complexity of the input model before deeper analysis. This leverages ATL's well-established efficiency in handling large state spaces and ensures a more tractable verification process.
- *Enhanced Expressiveness through NatATL*: The second phase ensures that the verification process fully accounts for bounded rationality constraints, enabling the analysis of both memoryless (nr) and recall-based (nR) strategies. This feature is particularly relevant in scenarios involving human-like decision-making, where strategies are shaped by cognitive limitations and behavioral heuristics.
- *Scalability for Complex Systems*: The framework's optimization techniques, including incremental strategy generation and symbolic representation, enable efficient scaling to accommodate real-world applications, from AI-driven decision-making to cooperative autonomous agents.

The framework operates in two sequential phases:

1. *Pre-processing with ATL*: The model undergoes an initial reduction using ATL's strategic operators, which prune invalid paths and produce a simplified representation containing only potentially satisfiable components.
2. *Verification with NatATL*: The refined model is then analyzed with *NatATL* to evaluate bounded strategies, ensuring both natural simplicity and cognitive feasibility in the decision-making process.

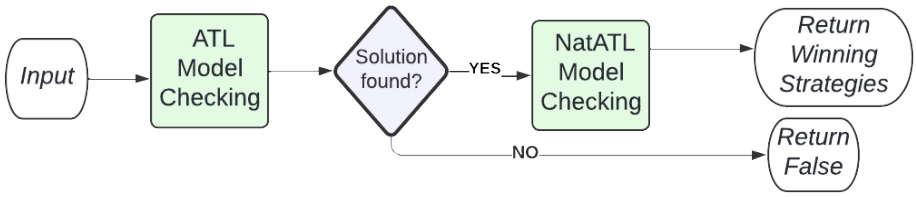

Fig. 2. Optimized *NatATL* Verification process

Fig. 2 illustrates the overall workflow, emphasizing the interplay between pre-processing and verification. The effectiveness of this optimization will be further analyzed in the experimental section, where we highlight the improvements in performance and verification efficiency enabled by this pre-processing feature.

4 EXPERIMENTS

We conducted extensive tests on the proposed verification tool using an HP Omen 15-ax213ng with an Intel i7-7700HQ 3.8 GHz CPU and 16 GB RAM.

Development was done in Python 3.9 via PyCharm, with GitHub as the hosting service. Since this tool integrates the first implemented *NatATL* verification module, we validated its accuracy against existing *ATL* model checking modules, already validated [21]. Table 1 summarizes the main features of *S4H* compared to other model checkers, emphasizing its novelty in supporting bounded rationality and recall-based strategies.

	MCMAS	MOCHA	S4H
Bounded Strategies	No	No	Yes
Memoryless Strategies	Yes	Yes	Yes
Recall-Based Strategies	No	No	Yes

Table 1. Comparison of Features Across Model Checkers

To initiate a preliminary experiment, it is crucial to examine the correspondence between the Boolean solutions generated by the two algorithms. So, each temporal operator was tested a hundred times to understand its behavior. Identical inputs were used for both *NatATL* and *ATL*, with their corresponding formula versions. For computational purposes, the *NatATL* complexity bound was increased to 10, considering that it is challenging for a human to track more than 10 guarded actions for a single strategy. In a comprehensive analysis of nearly a thousand tests, it was conclusively demonstrated that the *NatATL* verification system operates correctly. This is because *NatATL*, as an *ATL* update, will produce the same result for the same model. Therefore, this test is crucial in certifying the soundness of the implemented algorithm. Successively, to assess the time performance of our tool, we conducted 1000 more tests, systematically varying the number of states (i.e. model size), the complexity bound, and the number of agents within the formula coalition. Our analysis revealed that execution behavior is significantly influenced by the number of agents and the complexity bound. Each agent requires a specific set of strategies based on their potential actions, so increasing these values leads to longer execution times. For tests carried out with the recall tool, we set the maximum height of the structure at a predefined value of 5 for computational efficiency [3]. Thus, we observed slowdowns due to the necessity of representing histories. The execution time was found to be directly proportional to the degree of the tree, significantly influenced by the density of the transition matrix, particularly the number of different transitions from the same source state to the same destination state. Higher density led to longer execution times. In contrast, sparser matrices with lower tree degrees resulted in shorter execution times for this case. More specifically, we adopted two verification approaches: the first aimed at verifying model robustness, considering nondeterministic models where the matrix density was maximal and multiple occurrences of the same elements on the same row were

[3] Beyond a depth of 5, the tool started to experience slight delays in generating the entire tree, so we chose this height to maintain real-time processing.

present (hence nondeterminism). The second approach focused solely on strategy robustness, adopting classical incomplete models in multi-agent scenarios and discarding potentially invalid strategies for that non-robust model. The first approach empirically sets an upper bound on the temporal complexity due to the analysis of input models that consider all possible combinations of actions characterizing different transitions for each row (and for each element within it), addressing the worst-case scenario, which is less likely in practice. The second approach reflects the average case where the user generates an incomplete model due to scenarios in multi-agent systems, inevitably leading to lower and more efficient execution times compared to the worst-case scenario. The tables in Fig. 3, 4 summarize the time analysis conducted.

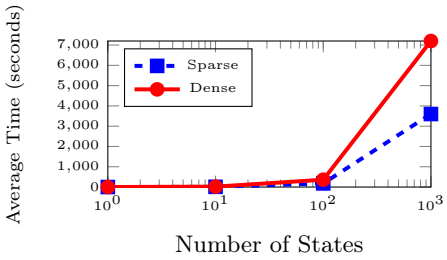

Fig. 3. Average case: Average time table comparison using dense and sparse transition matrix and $|A| = 3$

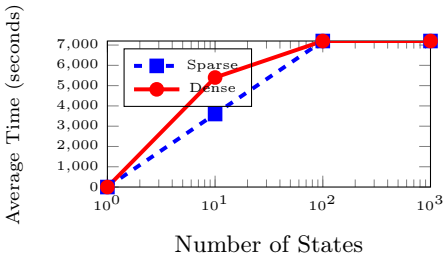

Fig. 4. Worst case: Average time table comparison using dense and sparse transition matrix and $|A| = 3$

4.1 Case Study: Logistics Robot System

As highlighted in the abstract, our work emphasizes the analysis of real-world models. However, the full-scale logistics robot model-comprising thousands of states and variables-would be unmanageable for both presentation and analysis. Therefore, before introducing the case study, it is important to note that we have developed a highly reduced and optimized version of the model. This reduction was achieved by systematically removing redundant and superfluous information while preserving the essential dynamics necessary for evaluating bounded strategic reasoning. Such a reduction not only provides a clear overview of the model configuration but also represents a significant advancement, as previous work did not address the synthesis and verification of natural strategies on real-world-scale models.

To further evaluate the practical applicability of *NatATL*, we conducted experiments on this reduced logistics robot system. The logistics domain is particularly relevant to AI-driven critical technologies due to its emphasis on efficiency, adaptability, and human-robot collaboration. This system comprises three agents: two robots (*ag1* and *ag2*) operating on conveyor belts, and a

third agent (*ag3*) responsible for auxiliary actions. The primary goal of these experiments was to assess the system's capability to maintain optimal performance under bounded strategy complexity constraints, thereby ensuring robust decision-making in dynamic and challenging environments.

This representative case study demonstrates that even under significant model reduction, the core challenges of bounded rationality in a logistics environment are effectively captured. The results underscore the potential of *NatATL* for handling complex, real-world applications in a scalable and efficient manner, marking a substantial step forward in the synthesis and verification of human-like strategies.

nr-Strategies in Logistics Robots In this model, the robots interact based on a state-transition graph where each state corresponds to a specific operational phase, such as moving between conveyor belts, picking up packages, and handling auxiliary actions. The *NatATL* framework is used to analyze whether a robot can achieve its goal of successfully completing logistics operations under bounded complexity constraints.

A memoryless strategy for a robot can be represented as a set of condition-action rules:

- If the robot is at the initial state and no task is assigned, remain idle.
- If a package is detected, move to the corresponding conveyor belt.
- If at the destination belt, execute the drop-off action.

As bounded complexity is quantified by the simplicity of conditions, at an initial state (s_0), where no task is assigned ($\neg assigned$), the robot remains idle until a package is detected. Once detected, the transition progresses through task execution states ($s_3 \rightarrow s_5$), ensuring smooth task completion while avoiding failure states (s_8). This approach validates usability and efficiency by maintaining system performance under limited computational resources.

nR-Strategies in Logistics Robots In more dynamic scenarios, recall-based strategies (*nR-strategies*) enhance decision-making by incorporating historical data. In the logistics domain, robots may need to adapt their actions based on past deliveries, congestion levels, or prior system failures. For example, a recall-based strategy might include:

- If a package has been reassigned multiple times in the last three steps, prioritize an alternative conveyor belt.
- If congestion was detected in previous states, adjust the movement pattern accordingly.
- Otherwise, proceed with standard task execution.

These strategies are formally expressed in *NatATL* using extended syntax to incorporate history-dependent rules while maintaining bounded complexity. For instance, an agent transitions from s_2 to s_5 (task completion state) by executing

STA,STI	SV*	MMI,VVB	MT*	STB,MVA
SVI,SVA	ST*	MTI,MTB	MV*	STA,MVB
MVI,MVA	MT*	STB,STI	SV*	STA,MVB
MTI,MTA	MV*	SVI,SVA	ST*	SVA,MTB
0	0	0	0	***

Table 2. Transition Matrix for Logistics Robot

a coordinated sequence over prior states ($s_0 \to s_1 \to s_2$). This enhances system adaptability in fluctuating environments.

The logistics robot case study exemplifies the necessity of bounded-rationality AI in mission-critical applications. By demonstrating how *NatATL* enables strategy verification in complex, resource-constrained environments, this study highlights its potential impact on AI-driven logistics, autonomous systems, and human-compatible robotics.

4.2 Empirical Validation of Optimization

As the reader may notice, the experiments focus on logistic robot study case, which tests the tool's ability to handle both nr-strategies and nR-strategies effectively. Our implementation approach brings several engineering advancements that directly impact its performance:

- *State Space Reduction*: The ATL pre-processing phase prunes unsatisfiable paths early, reducing the size of the CGS state space by up to 50% in benchmarks. This significantly mitigates the exponential growth typically associated with *NatATL*'s bounded strategy evaluation.
- *Incremental Strategy Generation*: Strategies are generated incrementally, prioritizing lower-complexity solutions and avoiding exhaustive enumeration. This approach reduces computational overhead without sacrificing accuracy.
- *Parallelized Pruning*: Parallelization of the pruning process during *NatATL* verification expedites the validation of strategies, particularly in scenarios involving recall-based strategies.
- *Symbolic Representation*: Both *ATL* and *NatATL* processes utilize symbolic representations of states and transitions, optimizing memory usage and improving scalability for large and complex models.

To evaluate our approach, Fig. 5, 6 present a comparison of average response times between traditional *NatATL* utilization and the combined $ATL{+}NatATL$ method both for nr and nR strategies. We tested unsatisfiable formulas (to analyze the worst computational scenario) with a fixed complexity bound and three agents involved. The practical viability of our tool is illustrated through blue line, representing the optimal response times for the $ATL{+}NatATL$ approach, compared to the red line, showing standalone *NatATL* usage. As expected, *ATL*

pre-processing filter significantly improves response times in real-world scenarios. Negative instances highlight the time differences, requiring *NatATL* to generate all possible strategies. Our new method filters real-time results without invoking the *NatATL* strategy generation algorithm, demonstrating the benefits of our optimization. Time constraints naturally increase with the number of agents and complexity bound, as the computational effort to generate individual strategies for each agent remains substantial.

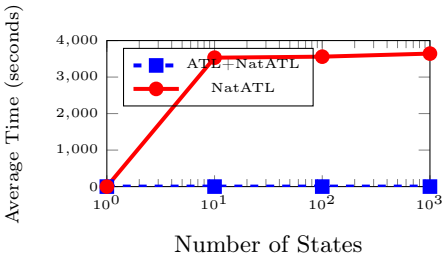

Fig. 5. *ATL* + *NatATL* times table: Memoryless approach

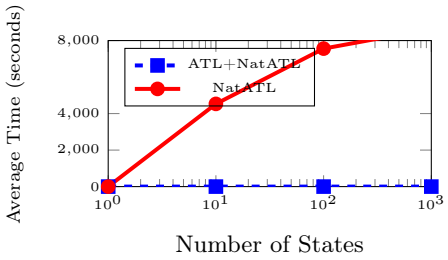

Fig. 6. *ATL* + *NatATL* times table: Recall approach

As the number of agents increases, the complexity bound limit needs to be reduced, indicating an implementation bottleneck due to the inherent logic. In fact, our pre-processing approach provides significant performance improvements, though careful management of complexity bounds is required as the number of agents increases. This demonstrates the practical effectiveness and limitations of our method in real-world applications.

4.3 Evaluation Metrics and Results

The performance of the tool was evaluated based on:

- **Accuracy:** Verifying that the tool produces correct results consistent with theoretical expectations.
- **Scalability:** Measuring execution times for models with varying sizes and complexity bounds.
- **Usability:** Assessing the ability of the tool to handle practical case studies effectively.

The results confirm that the tool works correctly, reproducing the expected results for all test cases. More specifically, the *ATL* pre-processing approach demonstrated significant efficiency improvements, reducing computation times by up to 90% compared to standalone *NatATL* verification for negative instances. Moreover, our dynamic strategy generation approach ensures the discovery of an optimal winning strategy for positive instances as soon as a solution is found.

5 CONCLUSIONS

In this paper, we have introduced *SH4*, a novel tool for the formal verification of bounded rationality in multi-agent systems by synthesizing human-like rational strategies. Our tool integrates bounded rationality, dynamically generating strategies for both memoryless (nr-strategies) and recall-based (nR-strategies) approaches, and optimizing through a pre-processing verification phase based on traditional *ATL* to significantly enhance computational efficiency and scalability, even in complex scenarios. Our experimental evaluation - including a detailed logistics robot system case study - demonstrates the practical applicability of our approach in real-world multi-agent environments. The results confirm that our optimization techniques effectively reduce computation time, particularly in heavy scenarios such as recall-based strategic reasoning. A particularly novel aspect of our work is the introduction of bounded strategy complexity as a criterion for distinguishing between human and artificial decision-making processes, thereby opening new avenues for assessing human-likeness in AI and offering a fresh perspective to the ongoing discussion surrounding the Turing Test.

Future Works. Our contributions pave the way for integrating this framework into Automated Turing Test applications and advancing neuroscience-inspired AI research. In particular, by setting an upper bound on the complexity of synthesized strategies - reflecting the inherent memory constraints of human cognition - it becomes possible to quantitatively assess the human-likeness of artificial agents. This integration would enable an Automated Turing Test where the ability of an agent to manage complex strategies within human cognitive limits serves as a proxy for its human-like decision-making. Additionally, future research may explore heuristic and machine learning-based enhancements to further optimize efficiency, especially for large state spaces and high complexity bounds. These extensions are expected to broaden the applicability of our framework in domains such as social robotics, human-in-the-loop systems, and beyond.

References

1. Alessandro Cimatti, Edmund Clarke, F.G.M.R.: Nusmv: A new symbolic model verifier. In: CAV 1999. pp. 495–499. Springer (1999)
2. Arrieta, A.B., Rodríguez, N.D., Ser, J.D., Bennetot, A., Tabik, S., Barbado, A., García, S., Gil-Lopez, S., Molina, D., Benjamins, R., Chatila, R., Herrera, F.: Explainable artificial intelligence (XAI): concepts, taxonomies, opportunities and challenges toward responsible AI. Inf. Fusion **58**, 82–115 (2020). https://doi.org/10.1016/J.INFFUS.2019.12.012, https://doi.org/10.1016/j.inffus.2019.12.012
3. Augier, M.: Simon says: Bounded rationality matters: Introduction and interview. Journal of Management Inquiry **10**(3), 268–275 (2001)
4. Breazeal, C.: Emotion and sociable humanoid robots. International journal of human-computer studies **59**(1-2), 119–155 (2003)

5. Ciardo, F., De Tommaso, D., Wykowska, A.: Human-like behavioral variability blurs the distinction between a human and a machine in a nonverbal turing test. Science Robotics **7**(68), eabo1241 (2022). https://doi.org/10.1126/scirobotics.abo1241
6. De Boer, L., Gaytan, J., Arroyo, P.: A satisficing model of outsourcing. Supply Chain Management: An International Journal **11**(5), 444–455 (2006)
7. Ferrando, A., Malvone, V., et al.: Hands-on vitamin: a compositional tool for model checking of multi-agent systems. In: Proceedings of the 25th Workshop" From Objects to Agents", Bard (Aosta), Italy. pp. 148–160 (2024)
8. Gerd Behrmann, Alexandre David, K.G.L.J.H.P.P.W.Y.M.H.: Uppaal 4.0 (2006)
9. Gigerenzer, G.: Bounded rationality: The adaptive toolbox (2002)
10. Gilboa, I., Samet, D.: Bounded versus unbounded rationality: The tyranny of the weak. Games and Economic Behavior **1**(3), 213–221 (1989)
11. Holzmann, G.J.: The model checker spin. IEEE Trans. Softw. Eng. **23**(5), 279–295 (1997)
12. Jamroga, W.: Logical Methods for Specification and Verification of Multi-Agent Systems. Institute of Computer Science, Polish Academy of Sciences (2015)
13. Jamroga, W., Malvone, V., Murano, A.: Reasoning about natural strategic ability. In: Larson, K., Winikoff, M., Das, S., Durfee, E.H. (eds.) Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems, AAMAS 2017, São Paulo, Brazil, May 8-12, 2017. pp. 714–722. ACM (2017), http://dl.acm.org/citation.cfm?id=3091227
14. Jamroga, W., Malvone, V., Murano, A.: Natural strategic ability. Artif. Intell. **277** (2019). https://doi.org/10.1016/J.ARTINT.2019.103170, https://doi.org/10.1016/j.artint.2019.103170
15. Jamroga, W., Malvone, V., Murano, A.: Natural strategic ability under imperfect information. In: Elkind, E., Veloso, M., Agmon, N., Taylor, M.E. (eds.) Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems, AAMAS '19, Montreal, QC, Canada, May 13-17, 2019. pp. 962–970. International Foundation for Autonomous Agents and Multiagent Systems (2019), http://dl.acm.org/citation.cfm?id=3331791
16. Jeremy Liang An Kong, Alessio Lomuscio, K.B.: Mcmas-dynamic (2016)
17. Lalla-Ruiz, E., Melián-Batista, B., Moreno-Vega, J.M.: Artificial intelligence hybrid heuristic based on tabu search for the dynamic berth allocation problem. Engineering Applications of Artificial Intelligence **25**(6), 1132–1141 (2012)
18. Langley, P., Laird, J.E., Rogers, S.: Cognitive architectures: Research issues and challenges. Cognitive Systems Research **10**(2), 141–160 (2009)
19. Lomuscio, A., Qu, H., Raimondi, F.: Mcmas: A model checker for the verification of multi-agent systems. In: CAV 2009. pp. 682–688. Springer (2009)
20. Lomuscio, Alessio, R.F.W.B.: Verification of the tesla protocol in mcmas-x. Fundam. Inform. **79**(3-4), 473–486 (2007)
21. Luongo, G.: A Quantitative Model Checking Tool for Multiagent Systems. Bachelor's thesis, University of Naples, Federico II (July 2023)
22. Nagat Drawel, Hongyang Qu, J.B.E.S.: Specification and automatic verification of trust-based multi-agent systems. Future Gener. Comput. Syst. **107**, 1047–1060 (2020)
23. Petersen, S.E., Sporns, O.: Brain networks and cognitive architectures. Neuron **88**(1), 207–219 (2015)
24. Rajeev Alur, Thomas A. Henzinger, F.Y.M.S.Q.S.K.R.S.T.: Mocha: Modularity in model checking. In: CAV 1998. pp. 521–525. Springer (1998)

25. Rajeev Alur, Thomas A. Henzinger, O.K.: Alternating-time temporal logic. J. ACM **49**(5), 672–713 (2002)
26. Stüttgen, P., Boatwright, P., Monroe, R.T.: A satisficing choice model. Marketing Science **31**(6), 878–899 (2012)
27. Zaji, A.H., Bonakdari, H.: Robustness lake water level prediction using the search heuristic-based artificial intelligence methods. ISH Journal of Hydraulic Engineering **25**(3), 316–324 (2019)