
# A Look Back at Strategy Logic



Citation for the original published paper (version of record):
Chatterjee, K., Henzinger, T., Piterman, N. (2026). A Look Back at Strategy Logic. Leibniz International Proceedings in Informatics, LIPIcs, 391.
http://dx.doi.org/10.4230/LIPIcs.CONCUR.2026.6

# A Look Back at Strategy Logic

**Krishnendu Chatterjee**
ISTA, Klosterneuburg, Austria

**Thomas A. Henzinger**
ISTA, Klosterneuburg, Austria

**Nir Piterman**
University of Gothenburg and Chalmers University of Technology, Gothenburg, Sweden

## Abstract

In this note, we recall the history and our motivation behind the development of Strategy Logic and we discuss some of the work that ensued from its introduction.

**2012 ACM Subject Classification** Theory of computation → Modal and temporal logics; Theory of computation → Verification by model checking; Theory of computation → Logic and verification; Theory of computation → Automata over infinite objects

**Keywords and phrases** Strategy Logic, Games, Automata

**Digital Object Identifier** 10.4230/LIPIcs.CONCUR.2026.6

**Category** Invited Contribution for the Test-of-Time Award

**Funding** *Krishnendu Chatterjee*: Supported in part by the Austrian Science Fund (FWF) COE grant 10.55776/COE12 and the ERC CoG 863818 (ForM-SMArt).
*Thomas A. Henzinger*: Supported in part by the ERC AdG 101020093 (VAMOS) and the FWF SFB grant F8502 (SPyCoDe).
*Nir Piterman*: Supported in part by the Swedish research council (VR) project no. 2025-05419 and the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation.

**Acknowledgements** We thank the jury that our paper introducing Strategy Logic at the Conference on Concurrency Theory (CONCUR) 2007 [10] was chosen for a CONCUR Test-of-Time Award at CONCUR 2026.

## From Temporal to Game Logics

Temporal logics have a wide range of applications, from the specification and analysis of reactive systems to planning in Artificial Intelligence. The most classical temporal logics are Linear-time Temporal Logic (LTL) [28] and Computation Tree Logic (CTL) [12], a branching-time temporal logic. Linear time specifies properties of sequential behavior; e.g., the LTL formula $\Diamond T$ denotes the set of trajectories that eventually reach a target proposition $T$. Branching time allows the existential and universal quantification over the trajectories of a system: the CTL formula $\exists\Diamond T$ (resp. $\forall\Diamond T$) asserts that some (resp. every) system trajectory reaches the target.

The alternating-time temporal logic ATL [1] introduced a quantification over the set of trajectories that result from the interaction of autonomous agents within a system: for a two-agent system, the ATL formula $\langle\langle 1\rangle\rangle\Diamond T$ asserts that agent 1 can choose its actions to ensure that the target is reached no matter which actions agent 2 chooses. Alternating time is adversarial, i.e., game-theoretic: the ATL formula $\langle\langle 1\rangle\rangle\Diamond T$ can be read as "player 1" having a strategy to reach the target against all strategies of "player 2." It is easy to see that alternating time generalizes branching time: the ATL formula $\langle\langle 1,2\rangle\rangle\Diamond T$ asserts that both agents together can ensure that the target is reached (against a nonexistent adversary), which


corresponds to existential quantification over strategies; the ATL formula $\langle\langle\rangle\rangle\Diamond T$ asserts that the target cannot be avoided no matter which actions the agents choose, which corresponds to universal quantification. As with classical temporal logics, several extensions of ATL have been studied in the literature, beginning with ATL$^*$ and the alternating-time $\mu$-calculus [1].

### Design Choices for Game Logics

Stateful systems with multiple agents are graph games, where the nodes of the graph represent system states, and the actions of the agents move a token from node to node, thus producing a system trajectory. In the study of graph games, and hence of logics for graph games, there are several important design choices: (a) the choice of the type of strategies the players can use, especially deterministic versus (more general) randomized strategies; (b) the choice of the type of game structures, especially turn-based versus (more general) concurrent game graphs; and (c) the choice of language for specifying the winning trajectories or quantitative objectives for the players. In randomized strategies, each player has access to a source of randomness; in concurrent games, the players can choose their actions simultaneously and independently; common quantitative objectives are mean-payoff or discounted-sum. These three design choices have important ramifications. For example, in two-player turn-based games with $\omega$-regular winning conditions, randomization does not add power: if a player can achieve a goal with a randomized strategy, they can achieve the same goal with a deterministic strategy. This is not the case for concurrent games: even in the simple "stone-paper-scissors" game, randomization gives players extra power. Both ATL and ATL$^*$ use LTL for specifying winning, but ATL$^*$ allows the unrestricted nesting of temporal operators within one strategic interaction while ATL restricts to a single temporal operator within one strategic interaction.

### Strategy Logic

Previous game logics such as ATL and ATL$^*$ or the alternating-time $\mu$-calculus reason about strictly competitive or strictly cooperative behaviors of agents; they do not capture other, non-zero-sum interactions studied in game theory. In particular, they are not able to express Nash equilibria or secure equilibria [9]. In a two-player game, a Nash equilibrium asks for a pair of strategies for the two players, such that no player can increase their payoff by deviating unilaterally from the strategy pair (in the case of qualitative objectives, a player receives payoff 1 if their winning condition is satisfied, and otherwise they receive payoff 0). In order to reason about equilibria, we needed a logic that can talk explicitly about strategies. The main conceptual novelty in our introduction of Strategy Logic [10, 11] was to consider strategies as first-order objects, with quantification over strategy variables. Strategy Logic generalizes ATL$^*$– e.g., the ATL$^*$ formula $\langle\langle 1\rangle\rangle W$, for an LTL formula $W$, corresponds to $\exists x_1.\forall x_2.\ W(x_1, x_2)$ in Strategy Logic – and can express Nash equilibria using the pattern

$$\exists x_1, x_2.\ (\forall y_1.\ W_1(y_1, x_2) \rightarrow W_1(x_1, x_2)) \wedge (\forall y_2.\ W_2(x_1, y_2) \rightarrow W_2(x_1, x_2)))\,,$$

for the winning LTL objectives $W_1$ and $W_2$ of the two players. In the original paper [10, 11], we focused on deterministic strategies and turn-based game graphs, and used LTL formulas for specifying winning objectives. We showed that Strategy Logic is incomparable in expressive power to the alternating time $\mu$-calculus (whose winning objectives go beyond LTL, but which cannot capture explicit strategy interactions), and that Strategy Logic is strictly less expressive than Monadic Second-Order Logic (MSO).

We also considered the model-checking problem for Strategy Logic. Inspired by landmark results such as Rabin's solution to Church's problem [30] and Pnueli and Rosner's solution to the problem of LTL synthesis [29], we encoded strategies as trees and used automata over

infinite trees as key tool for reasoning about Strategy Logic. This lead to a nonelementary upper bound for the model-checking problem of Strategy Logic. The nonelementarity arises, as it does for MSO (and for LTL with quantification over propositions), from quantifier alternations requiring the complementation (or nondeterminization [25, 15]) of tree automata. Using tree automata, we also identified several fragments of Strategy Logic whose model-checking complexity is more well-behaved. This includes the one-alternation fragment, which contains ATL* and can express interesting non-zero-sum properties such as Nash equilibria and secure equilibria. The specification complexity of model checking is 2EXPTIME-complete for both one-alternation Strategy Logic and ATL*, and their data complexity is P-complete.

The question of satisfiability for Strategy Logic remained open for a few years. It was settled negatively by [21], who showed that the satisfiability of Strategy Logic is undecidable. That paper also identified a fragment of Strategy Logic that generalizes ATL* for which satisfiability can be decided in 2EXPTIME.

### Design Choices for Strategy Logic

We introduced Strategy Logic with deterministic strategies over turn-based game graphs. These two choices were made for good reasons and are not independent. For turn-based games, randomized strategies do not add power. On the other hand, for concurrent games, deterministic strategies fail to capture the essence of simultaneous interaction between agents, and randomized strategies are needed even in simple scenarios such as "stone-paper-scissors." A restriction to deterministic strategies in concurrent game structures violates fundamental properties of such games, such as (a) determinacy (i.e., switching the quantifier order of players) in zero-sum games, or the existence of equilibria in matrix games, and (b) almost-sure or limit-sure winning for reachability objectives in graph games (where almost-sure and limit-sure winning represent the ability to win with probability 1, or with probability arbitrarily close to 1, respectively) [13]. Moreover, consider a concurrent game structure for two players and a quantifier (existential or universal) over deterministic strategies for player 1. This scenario corresponds to a turn-based game where player 1 first chooses an action and subsequently, given the choice of player 1, player 2 makes their choice; i.e., the "concurrent" game corresponds to a turn-based game on the product of the original state space and the actions of player 1 (cf. also [7]).

All this suggests the study of Strategy Logic for deterministic strategies over turn-based game graphs, or for randomized strategies over concurrent game graphs. While model checking the former turned out to be decidable (albeit nonelementary), the latter offers even less practical promise, as the existence of equilibria in concurrent games is undecidable [31, 8]. This is why we restricted our game structures to turn-based game graphs, which suffice to capture the essence of reasoning about deterministic strategies. In terms of winning objectives, we chose to concentrate on LTL as a convenient and familiar logical formalism.

### Extensions of Strategy Logic

Since the original paper [10, 11], several extensions of Strategy Logic have been considered in the literature. We first discuss some semantic extensions, to concurrent games, probabilistic aspects, and imperfect-information settings. The latter two extensions lead quickly to undecidability of the model-checking problem.

- *Concurrent games.* Strategy Logic was extended to concurrent game structures with strategies restricted to deterministic strategies [24]. Some confusion regarding the complexity of the corresponding model-checking problem arose initially, but later the same

authors established a nonelementary lower bound [22] to match their nonelementary upper bound. Their use of tree automata for model checking Strategy Logic over concurrent game structures validates our observation that for determinsitic strategies, turn-based games capture the essence of interaction. In addition, their nonelementary lower bound applies to turn-based game structures, and an alternative proof does not require strategy bindings [19]. Indeed, their matching lower and upper bounds link the complementation (or nondeterminization) of tree automata directly to the tower of exponentials that arises from quantifier alternation.

- *Probabilistic aspects.* The extension of Strategy Logic with probabilistic aspects, such as randomized strategies or probabilistic transitions in the game graph, was considered in [2]. Since Strategy Logic allows reasoning about non-zero-sum games, and basic questions related to equilibria with randomized strategies in concurrent games are undecidable, it is challenging to study such probabilistic settings in general. Decidability results can be obtained by a restriction to history-independent (a.k.a. memoryless) strategies.
- *Imperfect information.* Strategy Logic has also been extended to imperfect-information games, where each state is mapped to an observation and the players choose their actions dependent on the current history of observations (rather than the history of states) [5]. Imperfect-information games are related to the distributed-synthesis problem [26] and, therefore, undecidable even for deterministic strategies and turn-based game graphs. Decidability results can be obtained by considering hierarchical imperfect-information games [27].

Besides fragments of LTL for specifying winning in order to achieve better computational complexity results [4], several other syntactic variants of Strategy Logic have been considered. The following list gives some examples and makes no claim about completeness.

- *Equality of strategies.* The papers [24, 22] allow a strategy variable to be bound to multiple players. Thus, their formulas can force different players to use the same strategy.
- *Behavioral strategies.* In Strategy Logic, the satisfaction of a formula may require nonbehavioral strategies, which depend on the strategies of other players in nonrealizable ways (e.g., the current choice of player 1 may depend on a future choice of player 2). A syntactic fragment of Strategy Logic whose quantifiers can be restricted to behavioral strategies has been studied in [23].
- *Counting aspects.* Strategy Logic has been extended with graded strategy quantifiers, which count the number of satisfying strategy profiles [3], and with numerical bounds on temporal aspects of winning conditions and on the number of trajectories that satisfy a winning condition [16].

### Multi-agent Systems

Strategy Logic has been used extensively in research about Multi-Agent Systems (MAS), where many questions of correctness are naturally questions about strategic behavior. Indeed, several extensions of Strategy Logic have been motivated by problems from the MAS community, especially the need to reason about autonomous agents whose behavior is shaped by goals, incentives, and possible deviations. Closely related is the notion of *rationality* and its connection to game-theoretic concepts. The appeal of Strategy Logic in this setting is that quantification over strategies can state different rationality assumptions precisely. For example, "rational synthesis" considers several variations of the notion of equilibrium [17]. Furthermore, specifications of rationality in Strategy Logic can be algorithmically model checked and desired strategies automatically synthesized. Similar motivations arise in mechanism design and in the analysis of voting protocols, where the central issue is again whether self-interested agents have strategies that lead to desirable, stable, or secure outcomes [18, 20].

## Summary

Strategy Logic [10, 11] provides a powerful formal framework for reasoning about strategies in stateful games. The general logic is expressive and computationally expensive, and there are many interesting fragments and extensions – simple temporal fragments, behavorial fragments, probabilistic extensions, and extensions with hierarchical information and quantitative bounds – which aim to tailor the logic for better complexity or specific types of strategic reasoning. As agentic software becomes increasingly important, so does the role of formal methods in specifying, verifying, and synthesizing desired agent behavior. Strategies are the formalization of such behavior, which puts a logic that treats strategies as first-order objects at the center of the corresponding research.

The original question that motivated the development of Strategy Logic was to consider extensions of ATL$^*$ that can express Nash equilibria and similar non-zero-sum notions of game theory. Strategy Logic can do this within the fragment that allows only one quantifier alternation, for which model checking can be performed in doubly exponential time, as for ATL$^*$. However, it is not elegant to advocate a syntactic restriction for obtaining reasonable complexity. We still believe that there could be interesting logics in the space between ATL$^*$ and Strategy Logic which capture equilibria. One approach that comes to mind has the flavor of MSO with predicates [14, 6].

## References


1 Rajeev Alur, Thomas A. Henzinger, and Orna Kupferman. Alternating-time temporal logic. *Journal of the ACM*, 49(5):672–713, 2002. doi:10.1145/585265.585270.

2 Benjamin Aminof, Marta Kwiatkowska, Bastien Maubert, Aniello Murano, and Sasha Rubin. Probabilistic strategy logic. In *International Joint Conference on Artificial Intelligence (IJCAI)*, pages 32–38, 2019. doi:10.24963/ijcai.2019/5.

3 Benjamin Aminof, Vadim Malvone, Aniello Murano, and Sasha Rubin. Graded modalities in strategy logic. *Information and Computation*, 261:634–649, 2018. doi:10.1016/j.ic.2018.02.022.

4 Francesco Belardinelli, Wojciech Jamroga, Damian Kurpiewski, Vadim Malvone, and Aniello Murano. Strategy logic with simple goals: Tractable reasoning about strategies. In *International Joint Conference on Artificial Intelligence (IJCAI)*, pages 88–94, 2019. doi:10.24963/ijcai.2019/13.

5 Raphaël Berthon, Bastien Maubert, Aniello Murano, Sasha Rubin, and Moshe Y. Vardi. Strategy logic with imperfect information. *ACM Transactions on Computational Logic*, 22(1):5:1–5:51, 2021. doi:10.1145/3427955.

6 Mikolaj Bojanczyk. A bounding quantifier. In *Conference on Computer Science Logic (CSL)*, volume 3210 of *Lecture Notes in Computer Science*, pages 41–55. Springer, 2004. doi:10.1007/978-3-540-30124-0_7.

7 Patricia Bouyer, Romain Brenguier, Nicolas Markey, and Michael Ummels. Pure nash equilibria in concurrent deterministic games. *Logical Methods in Computer Science*, 11(2), 2015. doi:10.2168/LMCS-11(2:9)2015.

8 Patricia Bouyer, Nicolas Markey, and Daniel Stan. Mixed Nash equilibria in concurrent terminal-reward games. In *Conference on Foundation of Software Technology and Theoretical Computer Science (FSTTCS)*, volume 29 of *LIPIcs*, pages 351–363. Leibniz-Zentrum für Informatik, 2014. doi:10.4230/LIPIcs.FSTTCS.2014.351.

9 Krishnendu Chatterjee, Thomas A. Henzinger, and Marcin Jurdzinski. Games with secure equilibria. *Theoretical Computer Science*, 365(1-2):67–82, 2006. doi:10.1016/j.tcs.2006.07.032.

10 Krishnendu Chatterjee, Thomas A. Henzinger, and Nir Piterman. Strategy logic. In *Conference on Concurrency Theory (CONCUR)*, volume 4703 of *Lecture Notes in Computer Science*, pages 59–73. Springer, 2007. doi:10.1007/978-3-540-74407-8_5.

11 Krishnendu Chatterjee, Thomas A. Henzinger, and Nir Piterman. Strategy logic. *Information and Computation*, 208(6):677–693, 2010. doi:10.1016/j.ic.2009.07.004.

12 Edmund M. Clarke, E. Allen Emerson, and A. Prasad Sistla. Automatic verification of finite-state concurrent systems using temporal logic specifications. *ACM Transactions on Programming Languages and Systems*, 8(2):244–263, 1986. doi:10.1145/5397.5399.

13 Luca de Alfaro, Thomas A. Henzinger, and Orna Kupferman. Concurrent reachability games. *Theoretical Computer Science*, 386(3):188–217, 2007. doi:10.1016/j.tcs.2007.07.008.

14 Calvin C. Elgot and Michael O. Rabin. Decidability and undecidability of extensions of second (first) order theory of (generalized) successor. *Journal of Symbolic Logic*, 31(2):169–181, 1966. doi:10.2307/2269808.

15 E. Allen Emerson and Charanjit S. Jutla. Tree automata, mu-calculus, and determinacy. In *Symposium on Foundations of Computer Science (FOCS)*, pages 368–377. IEEE, 1991. doi:10.1109/SFCS.1991.185392.

16 Nathanaël Fijalkow, Bastien Maubert, Aniello Murano, and Sasha Rubin. Quantifying bounds in strategy logic. In *Conference on Computer Science Logic (CSL)*, volume 119 of *LIPIcs*, pages 23:1–23:23. Leibniz-Zentrum für Informatik, 2018. doi:10.4230/LIPIcs.CSL.2018.23.

17 Dana Fisman, Orna Kupferman, and Yoad Lustig. Rational synthesis. In *Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS)*, volume 6015 of *Lecture Notes in Computer Science*, pages 190–204. Springer, 2010. doi:10.1007/978-3-642-12002-2_16.

18 Wojciech Jamroga, Damian Kurpiewski, and Vadim Malvone. Natural strategic abilities in voting protocols. In *Workshop on Socio-Technical Aspects in Security and Trust (STAST)*, volume 12812 of *Lecture Notes in Computer Science*, pages 45–62. Springer, 2020. doi:10.1007/978-3-030-79318-0_3.

19 François Laroussinie and Nicolas Markey. Augmenting ATL with strategy contexts. *Information and Computation*, 245:98–123, 2015. doi:10.1016/j.ic.2014.12.020.

20 Munyque Mittelmann, Bastien Maubert, Aniello Murano, and Laurent Perrussel. Automated synthesis of mechanisms. In *International Joint Conference on Artificial Intelligence (IJCAI)*, pages 426–432. ijcai.org, 2022. doi:10.24963/ijcai.2022/61.

21 Fabio Mogavero, Aniello Murano, Giuseppe Perelli, and Moshe Y. Vardi. What makes ATL* decidable? A decidable fragment of strategy logic. In *Conference on Concurrency Theory (CONCUR)*, volume 7454 of *Lecture Notes in Computer Science*, pages 193–208. Springer, 2012. doi:10.1007/978-3-642-32940-1_15.

22 Fabio Mogavero, Aniello Murano, Giuseppe Perelli, and Moshe Y. Vardi. Reasoning about strategies: On the model-checking problem. *ACM Transactions on Computational Logic*, 15(4):34:1–34:47, 2014. doi:10.1145/2631917.

23 Fabio Mogavero, Aniello Murano, and Luigi Sauro. On the boundary of behavioral strategies. In *Symposium on Logic in Computer Science (LICS)*, pages 263–272. IEEE, 2013. doi:10.1109/LICS.2013.32.

24 Fabio Mogavero, Aniello Murano, and Moshe Y. Vardi. Reasoning about strategies. In *Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS)*, volume 8 of *LIPIcs*, pages 133–144. Leibniz-Zentrum für Informatik, 2010. doi:10.4230/LIPIcs.FSTTCS.2010.133.

25 David E. Muller and Paul E. Schupp. Alternating automata on infinite trees. *Theoretical Computer Science*, 54:267–276, 1987. doi:10.1016/0304-3975(87)90133-2.

26 Gary L. Peterson and John H. Reif. Multiple-person alternation. In *Symposium on Foundations of Computer Science (FOCS)*, pages 348–363. IEEE, 1979. doi:10.1109/SFCS.1979.25.

27 Gary L. Peterson, John H. Reif, and Salman Azhar. Lower bounds for multiplayer noncooperative games of incomplete information. *Computers and Mathematics with Applications*, 41:957–992, 2001. doi:10.1016/S0898-1221(00)00333-3.

28 Amir Pnueli. The temporal logic of programs. In *Symposium on Foundations of Computer Science (FOCS)*, pages 46–57. IEEE Computer Society, 1977. doi:10.1109/SFCS.1977.32.

29 Amir Pnueli and Roni Rosner. On the synthesis of a reactive module. In *Symposium on Principles of Programming Languages (POPL)*, pages 179–190. ACM, 1989. doi:10.1145/75277.75293.

30 M.O. Rabin. Automata on infinite objects and Church's problem. *American Mathematical Society*, 1972. doi:10.1090/cbms/013.

31 Michael Ummels and Dominik Wojtczak. The complexity of Nash equilibria in stochastic multiplayer games. *Logical Methods in Computer Science*, 7(3), 2011. doi:10.2168/LMCS-7(3:20)2011.