

A Translation of Probabilistic Event Calculus into Markov Decision Processes

Lyris Xu 

Dept. of Information Studies, University College London, UK

Fabio Aurelio D’Asaro 

Dip. di Studi Umanistici, Università del Salento, Lecce, Italy

Luke Dickens 

Dept. of Information Studies, University College London, UK

Abstract

Probabilistic Event Calculus (PEC) is a logical framework for reasoning about actions and their effects in uncertain environments, which enables the representation of probabilistic narratives and computation of temporal projections. The PEC formalism offers significant advantages in interpretability and expressiveness for narrative reasoning. However, it lacks mechanisms for goal-directed reasoning. Our work bridges this gap by developing a formal translation of PEC domains into Markov Decision Processes (MDPs), introducing the concept of “action-taking situations” to preserve PEC’s flexible action semantics. The resulting PEC-MDP formalism enables the extensive collection of algorithms and theoretical tools developed for MDPs to be applied to PEC’s interpretable narrative domains. We demonstrate how the translation supports both temporal reasoning tasks and objective-driven planning, with methods for mapping learned policies back into human-readable PEC representations, maintaining interpretability while extending PEC’s capabilities.

2012 ACM Subject Classification Computing methodologies → Knowledge representation and reasoning; Computing methodologies → Planning and scheduling; Computing methodologies → Temporal reasoning; Computing methodologies → Markov decision processes

Keywords and phrases Probabilistic Event Calculus, Markov Decision Processes, Temporal Projection, Narrative Reasoning

Digital Object Identifier 10.4230/LIPIcs.TIME.2025.21

Category Short Paper

Related Version *Full Version*: <https://arxiv.org/abs/2507.12989> [15]

Supplementary Material *Software*: <https://github.com/LyrisX02/PEC-MDP>
archived at `swb:1:dir:c557c388a86ca997275d269b6f53c804051a8727`

Funding *Fabio Aurelio D’Asaro*: Partially supported by the project FAIR – Future AI Research (PE00000013), under the NRRP MUR program funded by the NextGenerationEU.

1 Introduction

Reasoning about actions and their effects in dynamic, uncertain environments is a fundamental challenge in Artificial Intelligence (AI). The importance of narrative reasoning – the ability to represent and reason about sequences of events and their causal relationships over time – has been recognised since the early days of AI, leading to various temporal reasoning formalisms [10, 1, 5, 12]. Building on these early foundations, the Probabilistic Event Calculus (PEC) [3] has emerged as a powerful framework for representing and reasoning about uncertain scenarios. PEC extends the Event Calculus (EC) [5, 11], to provide an “action-language” style framework for modelling actions, their effects, and the evolution of world states over time under uncertainty. PEC-style frameworks offer highly interpretable



© Lyris Xu, Fabio Aurelio D’Asaro, and Luke Dickens;
licensed under Creative Commons License CC-BY 4.0

32nd International Symposium on Temporal Representation and Reasoning (TIME 2025).

Editors: Thierry Vidal and Przemysław Andrzej Wałęga; Article No. 21; pp. 21:1–21:5

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

and flexible representations of complex narratives, with demonstrated applications in domains such as medicine, environmental monitoring, and commonsense reasoning [2, 3]. Markov Decision Processes (MDPs), meanwhile, have established themselves as a powerful framework for modelling time-evolving systems controlled by an agent. MDPs and their variants are widely used across AI for decision-making under uncertainty and serve as the foundation for many statistical and reinforcement learning algorithms. While MDPs excel at control optimisation problems, PEC and its variants offer superior narrative interpretability but lack mechanisms for learning goal-directed behaviour.

A translation between these frameworks presents a compelling opportunity to bridge this gap, combining PEC’s human-readable representation with MDP’s computational efficiency and reinforcement learning capabilities. Such a bridge would allow for both efficient PEC implementation and the application of statistical learning techniques to narrative reasoning tasks. Towards this goal, we have developed a novel translation of PEC into an MDP framework, termed the PEC-MDP. This translation preserves the core assumptions and semantics of PEC while enabling the application of a wide range of MDP-based algorithms to PEC’s human-readable domains.

The key contributions of our work include:

- A formal translation of PEC domains to PEC-MDPs, with a Python implementation made available through a shared repository.
- An approach for performing temporal projection via the PEC-MDP formalism.
- A novel approach to planning under uncertainty in PEC domains.

2 Background

Probabilistic Event Calculus. A PEC domain $\mathcal{D}$ comprises a finite, non-empty set of fluents $\mathcal{F}$ and values $\mathcal{V}$, a finite set of actions $\mathcal{U}$, a set of fluent states $\tilde{\mathcal{S}}$, and a non-empty set of time instants $\mathcal{I}$. Fluents, as in classical narrative reasoning formalisms [12, 9], refer to properties of the world which may be affected by actions taken. A fluent state $\tilde{S} \in \tilde{\mathcal{S}}$ is an assignment of values to all fluents of the domain, while a partial fluent state $\tilde{X} \in \tilde{\mathcal{X}}$ is a subset of value-assigned fluents, i.e., $\tilde{X} \subseteq \tilde{S}$ for some $\tilde{S} \in \tilde{\mathcal{S}}$. PEC uses a set of action-language style propositions to specify probabilistic narrative information. A domain consists of:

1. A finite set of *v-propositions* which detail the values that a fluent may take.
 2. Exactly one *i-proposition* which specifies the probabilities of initial fluent states that hold at the minimum time instant.
 3. A finite set of *c-propositions* which model the causal effects of actions, each specifying a set of preconditions for effects to take hold (where at least one action is true), a partial fluent $\tilde{X}$ for each set of effects, and their corresponding probabilities.
 4. A set of *p-propositions* modelling the occurrence of actions, each specifying an action, a time instant, the probability for the action to be taken, and optional fluent preconditions.
- PEC supports a possible-worlds semantics, where a world is an evolution of an environment over time. Using this semantics, *temporal projection* computes the probability of fluent states or partial fluent states at future time points given an initial state distribution and a narrative of action occurrences. See [3] for more.

Markov Decision Processes. An MDP is defined as the tuple $(\mathcal{S}, \mathcal{A}, p_0, T, R)$, where $\mathcal{S}$ and $\mathcal{A}$ are finite sets of *states* and *actions* respectively (distinct from PEC states and actions). The initial state distribution is given by p_0 , where the probability of starting at time $t = 0$ in state s is given by $p_0(s)$. Transition dynamics are given by transition function

$T(s, a, s') = \Pr(s_{t+1} = s' | a_t = a, s_t = s)$, while the reward function $R(s, a, s') = r$ maps each state transitions to a numerical rewards. Behaviours in an MDP are encoded in a policy μ . Stationary policies map states to actions independent of time, either deterministically $\mu(s) = a$ or stochastically $\mu(s, a) = \Pr(a_t = a | s_t = s)$. Non-stationary policies [6] allow time-dependent mappings, μ_t , where if $t \neq t'$, then $\mu_t(s, a)$ can be different from $\mu_{t'}(s, a)$.

3 PEC-MDP Formalism

A translation of PEC domains into a MDP-derivative requires reconciliation of several key differences between the two frameworks: i. PEC dynamics operate on fluents (properties of the environment), while the MDP operates directly on environment states without internal structure; ii. PEC does not model rewards; iii. PEC assumes a progression of time independent to actions and environmental changes, while the MDP assumes a direct correspondence of each discrete time step to each episode of agent-state interaction; iv. PEC allows for simultaneous action-taking while the MDP does not; v. Action-taking in PEC conditions obligatorily on time-instants and optionally on specific partial fluent conditions, while the MDP's standard stationary policy which conditions only on the state an agent resides in.

Given that PEC's action-taking is conditioned on time, a non-stationary policy [6] is adopted to model time-dependent policies while maintaining stationary transition dynamics. The reward component of the MDP framework is omitted in the initial translation of PEC domains as PEC domains are without inherent reward signals. The PEC-MDP formalism is thus a reward-free MDP with a non-stationary policy.

To allow for efficient matrix operations, we translate PEC's natural language components into 0-based numerical encodings while maintaining bidirectional mappings to preserve interpretability. This encoding assigns each element an index based on arbitrary orderings over PEC fluents $\mathcal{F}$, values $\mathcal{V}$, and actions $\mathcal{U}$. The PEC-MDP state space is constructed through a two-step process: first, PEC fluent states are mapped to vector representations of fluent value indices (i.e. $(x_0, x_1, x_2, \dots)$ where x_j denotes that the index of the value taken by the fluent of index j); next, these vectors are mapped to integers for a more compact representation, acting as the base unit of a PEC-MDP state. The vector representation allows access to specific fluents, enabling two crucial functions: i. the mapping of a partial fluent state $\tilde{X}$ to a set of PEC-MDP states in which fluents in $\tilde{X}$ are entailed; ii. the update of a PEC-MDP state with the effect of a partial fluent state $\tilde{X}$ by modifying fluent elements affected by $\tilde{X}$ but retaining those which are not.

Next, to accommodate PEC's more flexible action-taking mechanisms into the MDP's rigid framework, we introduce action-taking situations composing *single*, *composite*, and *null* actions to simulate time-points at which agents perform one action, multiple actions simultaneously, or do not take any actions, respectively. The set of action-taking situations is determined by referring to p-propositions, to find all possible action combinations that may be taken at one time, including the null action situation where no action is taken.

Time instants are normalised to begin at 0 while preserving temporal ordering. The initial distribution is retrieved from PEC's i-propositions, as a probability distribution over PEC-MDP states mapped from fluent states. Transition probabilities are derived from c-propositions, where the effects of an action-taking situation are aggregated from its composite actions. The fluent state update operation associates a current state to an updated state given some effect $\tilde{X}$ with its corresponding probability. Finally, a non-stationary policy captures PEC's time-conditioned action probabilities from p-propositions, representing distributions over action-taking situations rather than individual actions.

The PEC-MDP translation has been fully implemented in Python and may be found [here](#). For a comprehensive overview of the functionalities, example domains, and usage instructions, readers are directed to the repository’s README file.

4 Applications

The PEC-MDP enables applying MDP techniques to PEC domains, most notably reinforcement learning for optimal decision-making, while preserving PEC’s original capability for temporal projection in narrative reasoning.

Previous implementations of temporal reasoning for PEC predominately calculate probabilities through summing over all possible worlds [3], or through approximate sampling [2]. Our proposed approach provides an exact solution through efficient matrix operations. We define policy-weighted transition matrices to propagate the initial distribution over PEC-MDP states forward to the queried time, to retrieve the distribution over states at that time. The probability that $\tilde{X}$ holds then is computed by summing probabilities over all fluent states that entail $\tilde{X}$. While a formal efficiency experiment has yet to be conducted, this method avoids the combinatorial explosion that comes with the computation of all possible worlds.

Next, to apply objective-directed strategies for the PEC-MDP, a desirability criterion must first be established in the form of an MDP reward function to guide agent behaviour. Once an appropriate quantitative reward signal is defined over outcomes, actions, or transitions, suitable reinforcement learning methods can be applied to discover optimal policies for the narrative domain. To preserve the interpretability advantages of PEC’s original formalism, learned policies may be translated back into human-readable p-propositions. This requires deterministic policies since action-taking situations must be separated into individual actions for p-propositions, meaning that probabilistic dependencies between actions cannot be preserved.

The mapping of a policy over action-taking situations is trivial where each action in a situation is performed at the corresponding instant where the fluent state holds. However, as this generates a large number of p-propositions, refinements can be applied to reduce this set for interpretability while maintaining semantic equivalence. These include eliminating p-propositions for unreachable state-time combinations and generalising fluent state conditions to their minimal distinguishing features.

The complete formal translation of the PEC-MDP may be found at [15], alongside a more detailed outline of temporal reasoning and objective-directed strategies.

5 Conclusion

Finally, let us note that other probabilistic extensions of the Event Calculus have been proposed, focusing on event recognition using probabilistic logic programming and learning from noisy data in both offline and online settings, i.e., [14, 13, 8, 4, 7]. In contrast, our PEC-MDP framework inherits the main features of PEC, which is more expressive (see [2] for a comparison), compiling it into an MDP. This shift lays the groundwork for reinforcement learning, positioning our work toward goal-driven reasoning and policy optimisation.

While our current work focuses on the PEC-MDP’s application to temporal projection and objective-directed learning, the PEC-MDP framework lays the groundwork for further applications that leverage MDP-based techniques within narrative domains. Beyond these broader applications, future work will include formal efficiency analyses and extending the framework to the Epistemic Probabilistic Event Calculus (EPEC) [2].

References

- 1 James F Allen. Maintaining knowledge about temporal intervals. *Communications of the ACM*, 26(11):832–843, 1983. doi:10.1145/182.358434.
- 2 Fabio Aurelio D'Asaro, Antonis Bikakis, Luke Dickens, and Rob Miller. Probabilistic reasoning about epistemic action narratives. *Artificial Intelligence*, 287:103352, 2020. doi:10.1016/J.ARTINT.2020.103352.
- 3 Fabio Aurelio D'Asaro, Antonis Bikakis, Luke Dickens, and Rob Miller. Foundations for a probabilistic event calculus. In *International Conference on Logic Programming and Nonmonotonic Reasoning*, pages 57–63. Springer, 2017. doi:10.1007/978-3-319-61660-5_7.
- 4 Nikos Katzouris, Georgios Paliouras, and Alexander Artikis. Online learning probabilistic event calculus theories in answer set programming. *Theory and Practice of Logic Programming*, 23(2):362–386, 2023. doi:10.1017/S1471068421000107.
- 5 Robert Kowalski and Marek Sergot. A logic-based calculus of events. *New generation computing*, 4:67–95, 1986. doi:10.1007/BF03037383.
- 6 Erwan Lecarpentier and Emmanuel Rachelson. Non-stationary markov decision processes, a worst-case approach using model-based reinforcement learning. *Advances in neural information processing systems*, 32, 2019. URL: <https://dl.acm.org/doi/10.5555/3454287.3454935>.
- 7 Periklis Mantenoglou. *Reasoning over Complex Temporal Specifications and Noisy Data Streams*. PhD thesis, Univeristy of Athens, 2024.
- 8 Periklis Mantenoglou, Alexander Artikis, and Georgios Paliouras. Online event recognition over noisy data streams. *International Journal of Approximate Reasoning*, 161:108993, 2023. doi:10.1016/J.IJAR.2023.108993.
- 9 John McCarthy. Programs with common sense, 1959.
- 10 John McCarthy and Patrick Hayes. Some philosophical problems from the standpoint of artificial intelligence. In B. Meltzer and Donald Michie, editors, *Machine Intelligence 4*, pages 463–502. Edinburgh University Press, 1969.
- 11 Rob Miller and Murray Shanahan. Some alternative formulations of the event calculus. In *Computational Logic: Logic Programming and Beyond: Essays in Honour of Robert A. Kowalski Part II*, pages 452–490. Springer, 2002. doi:10.1007/3-540-45632-5_17.
- 12 Erick Sandewall. Features and fluents: A systematic approach to the representation of knowledge about dynamical systems. *Technical Report LiTH-IDA-R-92-30, Department of Computer and Information Science*, 1992.
- 13 Anastasios Skarlatidis, Alexander Artikis, Jason Filippou, and Georgios Paliouras. A Probabilistic Logic Programming Event Calculus. *Theory and Practice of Logic Programming*, 15:213–245, March 2015. doi:10.1017/S1471068413000690.
- 14 Anastasios Skarlatidis, Georgios Paliouras, Alexander Artikis, and George A. Vouros. Probabilistic Event Calculus for Event Recognition. *ACM Transactions on Computational Logic (TOCL)*, 16(2):11, 2015. doi:10.1145/2699916.
- 15 Lyris Xu, Fabio Aurelio D'Asaro, and Luke Dickens. A translation of probabilistic event calculus into markov decision processes, 2025. arXiv:2507.12989.