Machine Learning (2022) 111:1765–1797
https://doi.org/10.1007/s10994-022-06130-x

Semi-Lipschitz functions and machine learning for discrete dynamical systems on graphs

H. Falciani[1] · E. A. Sánchez-Pérez[1]

Received: 5 September 2020 / Revised: 25 January 2022 / Accepted: 6 February 2022 /
Published online: 23 March 2022

Abstract
Consider a directed tree $\mathcal{U}$ and the space of all finite walks on it endowed with a quasi-pseudo-metric—the space of the strategies $\mathcal{S}$ on the graph,—which represent the possible changes in the evolution of a dynamical system over time. Consider a reward function acting in a subset $\mathcal{S}_0 \subset \mathcal{S}$ which measures the success. Using well-known facts of the theory of semi-Lipschitz functions in quasi-pseudo-metric spaces, we extend the reward function to the whole space $\mathcal{S}$. We obtain in this way an oracle function, which gives a forecast of the reward function for the elements of $\mathcal{S}$, that is, an estimate of the degree of success for any given strategy. After explaining the fundamental properties of a specific quasi-pseudo-metric that we define for the (graph) trees (the bifurcation quasi-pseudo-metric), we focus our attention on analyzing how this structure can be used to represent dynamical systems on graphs. We begin the explanation of the method with a simple example, which is proposed as a reference point for which some variants and successive generalizations are consecutively shown. The main objective is to explain the role of the lack of symmetry of quasi-metrics in our proposal: the irreversibility of dynamical processes is reflected in the asymmetry of their definition.

Keywords Graph distance · quasi-pseudo-metric · reinforcement learning · Lipschitz function · bifurcation metric

Editor: Pradeep Ravikumar.

E. A. Sánchez-Pérez
easancpe@mat.upv.es

H. Falciani
herfal@upvnet.upv.es

[1] Instituto Universitario de Matemática Pura y Aplicada, Universitat Politècnica de València, Camino de Vera s/n, 46022 Valencia, Spain

1 Introduction

Artificial intelligence tools use fundamental mathematical objects as structural support for the learning algorithms. Although the forms in which they are presented and applied are varied and flexible, machine learning (ML) algorithms and, in particular, reinforcement learning (RL) methods rely on strict formal structures. Often the vector representation of the data is used, which facilitates the information items to be considered as elements of normed spaces. When a distance is needed for the construction of the algorithm, a natural option is to choose a norm. The Euclidean distance is the first candidate (Kubat, 2017); the reader can find an explanation in Sect. 3.2 of this work. This is, for example, a classic way of dealing with the problem of text analysis, in the context of vectorial representation of meaning, which lies in the idea that the words in a text can be associated with the vectors of a normalized space—vector embedding,—and the words that share a similar context should be close in the vector space—measured with the norm—, which means that they have similar semantics (Camacho-Collados & Pilehvar, 2018).

Similarity functions, which need not in general be distances, can also be used to quantify metric-like notions. Cosine similarity is a canonical example of such a tool (see Xia et al., 2015) and references therein). In these cases, it is used as a representation technique, an embedding of the data set in a normed space, where the distance associated to the norm is the only structural feature that is intended to be preserved. When the relations among the elements of the data set are better understood—for example, when addition of vectors or product by scalars are introduced in the model,—the representation can be enriched and the results improve accordingly.

However, these classical frameworks are not the only suitable formal supports for the models. Other mathematical structures than normed spaces are used as well to support machine learning models (Bronstein et al., 2017). Graph theory provides such a theoretical support, and there are a lot developments that explote the possibilities that this fundamental framework provides, specially when metric tools are added to the underlying structure (Buckley & Harary, 1990; Graham et al., 1977; Hakimi & Yau, 1965). The reader can find some of them in the papers by (Cao et al., 2012, 2016)—metric learning— (Chami et al., 2020; Driessens et al., 2006)—reinforcement learning—(Kyng et al., 2015)—Lipschitz learning;—updated surveys on these topics can be found in the references (Goyal & Ferrara, 2018) and (Nickel et al., 2015). However, combining the graph structure with compatible topological tools is not as common. In particular, although metrics on graphs are often used, quasi-metrics for finding similarities in datasets have only recently been considered. As explained in the Introduction of the work by Zhang et al. (2019), it is usual to consider both metric spaces (Chávez et al., 2001; Hjaltason & Samet, 2003) and graphs (Wu et al., 2016), but the tools known in these contexts seem to be inefficient when considered together. The framework provided by the so-called attribute graphs (Xu et al., 2012) could be considered in a sense to be similar to the graph/quasi-metric approach, but they differ in some crucial points (Zhang et al., 2019).

Let us mention that there is a rich literature on the topic from which we start our construction: graph-based metric spaces. Most of these topological structures were created for supporting resolution methods to applied problems, and is not strange to find new metrics and ideas that were primarily introduced in an applied contexts, as chemistry and other fields, see for example (Bu et al., 2014; Chebotarev 2011; Klein and Randić 1993), and of course in artificial intelligence (Bunke & Shearer, 1998; Chen & Safro, 2011; Gao et al., 2010). In fact, this is a well-known classical setting Entringer et al. (1976) for the development of new network analysis methods (Barnes & Harary, 1983; Brandes, 2005; Goddard & Oellermann, 2011), including fundamental tools for solving nowadays classical routing problems (Bellman, 1958; Goldberg, 1993).

Thus, in this paper we are interested in considering a special structure for the set of items/states of the system (a graph) that can both be connected with the dynamic character of the actions on the set (i.e. discrete dynamical systems) and with a causal relation among them, for example dependency graphs (Barrett et al., 2004). Note that the nomenclature (states, actions,...), and some aspects of our formalism have been adopted from some classical dynamical systems and machine learning environments. Thus, we propose an asymmetric topological approach for reinforced learning on graph-based structures. Walks on these graphs can be considered as elements of a new quasi-metric space, in which the graph impose some structural constrains on the topological properties. Taking into account that we are interested in models for dynamical systems, we will center our attention on acyclic connected and directed graphs, often called polytrees or simply trees (Bondy & Murty, 1976).

Our main idea is that *quasi-metrics—more than metrics—allow us to consider the intrinsic asymmetry that underlies the graphs used as models of evolutionary processes.* As time goes by, the directionality at the edges of the associated graphs—arrows—, becomes the main property, and the notions of the metric type used in the model have to represent this asymmetry. The consecutive (time) step between two states v_1 and v_2 of a system can be represented by an arrow from v_1 to v_2 in the graph model. Thus, the distance from v_1 to v_2 can be small—the dynamic process goes in this direction,—while returning from v_2 to v_1 should be impossible if, for example, the process is not reversible, and therefore the distance should be infinite.

This characteristic asymmetry in the definition of metric tools is fundamental to introduce the notion of quasi-metrics. Indeed, the main property of the quasi-metric versus the metric is exactly that $q(v_1, v_2)$ does not necessarily coincide with $q(v_2, v_1)$. That is, it is not assumed that one of the three axioms of a definition of a metric—symmetry–is maintained. The main objective of this paper is how to introduce this metric asymmetry into graph-based models, and to show how this can be used. To illustrate our ideas we will develop a particular framework: polytrees endowed with bifurcation quasi-metrics.

The second fundamental part of the problem we face in the paper is its dynamic nature. We are interested in metric graphs, but not considered as static structures but as dynamic ones. That is, we want to analyze how a dynamic (discrete-time) system can be represented using metric notions on a graph, since the ultimate goal is to demonstrate that non-symmetric distances are suitable tools to build automatic decision making algorithms that provide reinforcement learning tools. Consequently, the main reference we find in the dynamical systems research is the framework of the Markov Decission Proccesses (MDPs). Already in the early literature regarding this topi (Bellman, 1957; Howard, 1960), metric notions where used as fundamental tools (Puterman, 1994). The fact of the non-symmetric nature of the evolution processes is well-known and has been introduced in the context of the Markov Decission Processes (MDPs) research at all levels. For example, in multiagent MDPs, to introduce this fact becomes a basic instrument to reduce the size of the st of interactions between agents (Dolgov & Durfee, 2006), see also Boutilier (1999) and Singh and Cohn (1998). As we will explain later on, the framework of the MDPs is closely related to our in what respects the dynamic nature of the processes we want to model, and reinforcement learning algorithms could be designed using non-symmetric distances (quasi-metrics). In this paper we intend to show with simple examples how this asymmetry in the measurement of distances can provide a fruitful context, instead of the absolute notions of asymmetry that are commonly used in graphs, mainly the directionality of arrows in directed graphs. Simplistically, MDPs can be viewed as dynamic Bayesian networks (a dynamic probabilistic directed acyclic graph). The non-symmetrical distance can help to relax the assumption about the directionality of the graph by allowing some gradation in its description.

Concretely, the elements of our models are finite directed walks on the graph, that we call strategies. A reward function is then defined to evaluate the best solutions to the problem, and its extension—the oracle function—will give the main tool for the forecast on the dynamical system. As predictive tool we will use the extension expressions provided by the classical formulas of McShane and Whitney, adequately adapted for the case of semi-Lipschitz functions on quasi-pseudo-metric spaces. Our aim is to provide new models for understanding success reward systems in which there is an intrinsic asymmetry in the definition of the metric, and where the items of the systems are represented as walks in a graph. This is the case, for example, when we want to introduce a directionality given by the time variable in evolutionary systems, or the case when the topology must reflect a hierarchical relation among the nodes of the graph. As we said, the natural underlying metric structure in a graph representing a dynamical system is not symmetric: non-symmetric distances model in an optimal way the evolution of the system with the time. It must be said that a (symmetric) distance is often used for non-directed graphs: the so called path distance, defined as the infimum of the number of edges (maybe weighted) in all possible paths that connect different nodes of a graph; it can be found for example in the book by Deza and Deza (2009). The experienced reader knows that this kind of structure is conceptually related to the theoretical setting of the bayesian networks, that use these constructions as models, and also with the so called Markov networks. Although the relationship with these approaches is evident, we must indicate that our mathematical framework is not the same, since we are not using probabilistic arguments, as the reader will notice. Our method could allow to model two problems that we are interested in, and are often solved using the mentioned tools: causal graphs and paths in time evolutionary systems (Chen & Giménez, 2010; Casteigts et al., 2011) (see also the references therein).

Once the basic structure of the system is defined—an underlying directed tree, a set of strategies, a quasi-pseudo-metric on it,—we use classical extension formulas for Lipschitz functions on (quasi-pseudo)-metric spaces as main tools for the construction of our learning algorithms. The definition of an adequate reward function on a training set of strategies S_0 together with an extension rule will give an estimate of the oracle function $\mathcal{O}$ for all the strategies in S. The increase of the training set S_0 would leave to a better estimate for $\mathcal{O}$, constituting in this way a reinforcement learning algorithm that will be checked in the last section of the paper. The increase in the set of evaluated successful paths (which are included in S_0 at each step) plays the role of the cumulative reward that is usually given by the value function in RL algorithms. We have already used this method in general metric spaces in the applied context of the prospective in financial markets (Calabuig et al., 2020).

An example of how our results can help is provided by the increasing interest of the so called graph embedding algorithms, which embed a graph into a vector space. The structure and the main properties of the graph are preserved in this embedding, but some properties, as the asymmetric transitivity in the graph inherent to directed graphs cannot be adequately represented. Some new ideas have been already proposed to solve this problem (Ou et al., 2016), which could be also faced by introducing non-symmetric distances, which also allows to correctly model transitivity properties. Our ideas can also help as theoretical framework for other current problems in deep reinforcement learning (Sutton & Barto, 2017), that often uses graph-based representations as operational tool (Waradpande et al., 2020).

As we have said, the most similar framework to the one we propose here are Markov Decision Processes, and the evolved versions of these in reinforcement learning (such as Q-learning) constitute the most similar reference that can currently be found in the scientific literature. Indeed, the contextual framework of the proposed ideas is the one for which the Markov decision processes applies. The notions of state —the current situation of the agent—, and actions that define the dynamics of the process, are the main elements that are considered. Also the concept of reward assigned to each action, that supports the decision of the consecutive steps of the proccess, is key in our formalism. So, in some sense, Q-learning methods and updated current procedures based on the same principles are the natural methodological partners of our method. However, the way we face the problem is opposite to the one in which Markov processes are performed: while some stochastic notions are inherent to our arguments too, the definition of the quasi-metric space from which the extension function provides the tool for decision making is deterministic, and the constitution of the metric space from which we extend is absolutely history-dependent. Indeed, the size of the quasi-metric space is increasing while the process evolves, but all the "experience" accumulated in the previous steps are equally considered for proposing the next action on the system. This will be clear in the examples we provide at the end of the paper. The interested reader can find an exhaustive explanation of Markov Decision Processes and their applications as a reinforcement learning method in Sigaud and Oe (2013). But the main references about similar general working philosophy are more connected to recent developments on history-dependent adaptation of Q-learning (POMDPs,...), and in general, as we already said, MDPs related methods. The articles by He et al. (2020); Daswani et al. (2013); Majeed and Hutter (2018) and the references therein provide examples of this methodological approach. However, as we said before, our method is based on other ideas: non-symmetric topological structures and semi-Lipschitz extensions for history-dependent reinforcement learning.

The structure of the paper is as follows. After the explanation in Sect. 2 of the basic topological and graph-theoretical structures, we define suitable quasi-pseudo-metrics on graphs in order to represent appropriate topologies for these spaces. In the first part we analyze some options for defining quasi-pseudo-metrics that would be useful for modeling graphs constructed over sets of action sequences on a given system. The second step consists of the development of the extension method for the reward function acting in the space. To do so, we follow some theoretical tools related to the ones developed by (Mustăţa, 2001, 2002). Specifically, we analyze the minimal extensions provided by the quasi-metric version of the McShane and the Whitney extension formulas for (semi)-Lipschitz maps. Interpolation among these situations could be interesting candidates for meaningful extensions of Lipschitz maps, which will be the starting point of our method. We show all these elements in Sect. 3. After presenting the main theoretical results, we will show some situations in which our ideas can be set. The first one deals with a problem that involves sequences of states and actions in a particular decision system, which we call the "problem of the drunk man crossing a bridge" (DBProblem) (Sect. 4). Sect. 4.2 is devoted to analyze in an applied context the properties of the extension functions, as cornerstones of the algorithm that provides the decision of the system. In Sect. 6, a specific example of application of non-symmetric distances is explained, and some relevant aspects as efficiency or scalability of the algorithm are discussed. This example has been designed to show that asymmetry is crucial for solving the problem: proper quasi-metrics—and no metrics—, are used, showing the advantages of these tools instead. A final section with the main conclusions of the paper is also included.

2 Preliminaries

In this paper we will analyze reward functions defined on the (quasi-pseudo-metric) graphs that model the original structured system. The method proposed provides a forecast for strategy success. This is done using a Lipschitz extension method given by a convex combination of the McShane and the Whitney extensions of real functions defined on subspaces of (quasi-pseudo-)metric spaces. Roughly speaking, after fixing the graph that models the problem, we have to find a (quasi-)metric subspace of strategies on the graph for which the reward function is known. After computing its Lipschitz constant, we apply both the McShane and the Whitney formulas for extending the reward function to the whole quasi-pseudo-metric space of strategies; quasi-pseudo-metric versions of such formulas will be needed. We will call that extension an oracle function, that can be used to design new prediction tools for graph analysis.

Probably the most widespread tool for dealing with this type of problems are non-history dependent methods based on the assumptions of Markov processes. Markov decision processes, as well as Q-learning methods, are typical sources of algorithms for this type of analysis. However, our approach is radically different: essentially, only metric notions are considered for the extension of the value and reward functions, and all the experience accumulated during the evolution process is used to obtain the best extension at a given step. The probabilistic character of the last steps is only introduced at the end, using some basic Monte Carlo type arguments. The interested reader can find a full explanation of these non-history-dependent methods in Sigaud and Oe (2013).

The main difference with other approaches that use the extension of functions acting in metric spaces, is that we are introducing non-symmetric distances instead of conventional metrics. The reason is that for the modeling of evolutionary processes it is convenient to take into account that in general they are not reversible, so distances that "look at the past" cannot behave as if they "look at the future". Quasi-metrics are perfect tools to model this fact, as we intend to explain in this paper.

Thus, there are some recently published papers in which some related (metric) ideas can be found. (Dong et al. 2018) propose a technology based on the use of Lipschitz extensions of (real) functions acting in metric spaces; see also the papers by (Gottlieb et al., 2014; Uv and Bousquet, 2004; Jia et al., 2015). Indeed, nowadays there is an increasing interest in the study of minimal extensions of Lipschitz maps defined on graphs. This topic was originally studied in the context of the development of new methods related to reinforcement learning. In fact, it started to be used in metric learning at the end of the XX Century, and nowadays has become a relevant tool in metric learning, as can be seen in the work of (Dong et al., 2018; Shaw et al., 2011) (see also the references therein). The reader can find an updated survey on the topic in the first chapter of the work of (Rao, 2015), and in the papers of (Driessens et al., 2006; N'Guyen et al., 2013). However, the framework is restricted to Lipschitz extensions of functions acting in (symmetric) metric graphs, while our theoretical approach is essentially non-symmetric. Moreover, the most used distances come from standard norms defined on finite dimensional linear spaces, but this is not the only way of introducing metrics in machine learning. For example, Mahalanobis distances instead of Euclidean distances are often used for performing a distance learning approach, assuming an underlying linear structure in the representation. However, Lipschitz continuity on general metric spaces—with no linear structure—has also being applied to reinforced learning recently by (Asadi et al., 2018); related developments are referred also in this work. Since the problems that motivate our research are non-symmetric, metrics must be substituted by quasi-metrics for the aim of constructing forecasting methods. In this direction —and in order to give conceptual support to our research,—we point out that the theoretical setting for extending semi-Lipschitz maps in quasi-metric spaces have been investigated in recent years (Mustăţa, 2001, 2002; Romaguera & Sanchis, 2000). This opens the door to extend the application of the Lipschitz extension technique to this non-symmetric context.

Let us introduce now some formal definitions that will be needed. As usual, we will use the words "metric" and "distance" interchangeably. A quasi-pseudo-metric is a function $d : M \times M \to \mathbb{R}^+$, where M is a set and such that for $a, b, c \in M$,

1. $d(a, b) = 0$ if $a = b$, and
2. $d(a, b) \leq d(a, c) + d(c, b)$.

($\mathbb{R}^+$ is the set of non-negative real numbers.) Such a function is enough for defining a topology by means of the basis of neighborhoods that is given by the open balls. If $\varepsilon > 0$, we define the ball of radius $\varepsilon > 0$ and center in $a \in M$ as

$$B_\varepsilon(a) := \Big\{ b \in M : d(a, b) < \varepsilon \Big\}.$$

This topology is given by the (countable) base of neighborhoods provided by the balls $B_{1/n}(x) = \{y \in X : d(x, y) \leq 1/n\}, n \in \mathbb{N}$. The resulting metric/topological structure (M, d) is called a quasi-pseudo-metric space.

If $d(a, b) = d(b, a)$ for $a, b \in M$, then it is called a pseudo-metric. If $d(a, b) = 0$ only in the case that $a = b$ it is called a quasi-metric, and if both requirements are satisfied, d is called a metric or a distance. The function d^{-1}, given by

$$d^{-1}(a, b) := d(b, a), \quad a, b \in M,$$

can also be defined: it is also a quasi-pseudo-metric, that is called the conjugate quasi-pseudo-metric. If d is a quasi-metric, we have also that

$$\max\{d(a, b), d^{-1}(a, b)\}, \quad a, b \in M,$$

is a metric.

The set of real Lipschitz functions acting in a metric space M is defined by such functions $f : M \to \mathbb{R}$ that satisfy that

$$|f(a) - f(b)| \leq K d(a, b).$$

The Lipschitz constant of f is the infimum of all the constants K satisfying the inequality.

The reader can find all the information on Lipschitz functions that is needed in the book by Cobzaş et al. (2019). The problem of extending Lipschitz functions acting in subsets of graphs has been recently considered, both from the theoretical and the computational points of view (Kyng et al., 2015; Rao, 2015). In this research the attention is centered in the case in which the topology defined on the graph is given by a (strict) metric, in particular a weighted graph distance. A weight—a positive real number—is given to each edge of the graph; then the distance among the nodes a and b of the graph is defined as the infimum of all the sums of weights associated to each path that connect a and b. In this paper we introduce different topologies on the graph that are defined by non-symmetric functions—that is, by quasi-pseudo-metrics—not necessarily defined in this way. Moreover, the research presented by Kyng et al. (2015) and Rao (2015) do not use the McShane—or the Whitney—formula(s), since they are interested in finding extensions with higher order of smoothness.

The main tools that we use in our development is the extension of Lipschitz functions acting in subspaces of a metric space to the whole metric space. We will consider the non-symmetric version. A classical result of the mathematical analysis establishes that this can always be done: indeed, the McShane-Whitney theorem says that if B is a (metric) subspace of a metric space (M, d) and $T : B \to \mathbb{R}$ is a Lipschitz function with Lipschitz constant K, there is always a Lipschitz function $\tilde{T} : M \to \mathbb{R}$ extending T and with the same Lipschitz constant: that is, $\tilde{T}(a) = T(a)$ for all $a \in B$.

In particular, the function

$$T^M(x) := \sup_{a\in B}\{T(a) - K\, d(x,a)\}, \quad x \in M,$$

that is called the McShane formula, provides such an extension. The Whitney formula, given by

$$T^W(x) := \inf_{a\in B}\{T(a) + K\, d(x,a)\}, \quad x \in M,$$

provides a different extension. We will use the as constructive tools for our approximation.

We have used the Lipschitz extension method for (symmetric) distances in a recent paper, in which a reinforcement learning procedure for time series problems is presented Calabuig et al. (2020). The problem proposed there is essentially different in that it is based on a vector-valued representation and a real function defined on it, but the use of Lipschitz extensions appears in it. As we already explained, this is our methodological purpose and can be applied in a wide class of different problems. In the cited paper, the comparison is made with neural networks, which are a standard technique that can be directly applied to the proposed problem. However, the situation in the case at hand (the present work) is different: the theoretical foundations needed are somewhat more complicated, since we need to clearly define the element space as a graph-quasi-metric space, and the action space as a similar space with some kind of duality with the previous one. New definitions of quasi-metrics and the proof of the essential facts concerning them are also needed, as well as some examples to illustrate the setting.

So, we are interested in using such extensions for real functions acting in quasi-pseudo-metric spaces. The theoretical results extending the McShane-Whitney theorem to this situation are easy to prove in the same way that the original theorem, and were essentially presented by Mustăţa (2001, 2002). As far as we know, these are the earliest references for these results. But the asymmetry of the quasi-metric functions change the results, that have to be rewritten. We write the main result below for the aim of completeness. New definitions are needed.

Let (S, q) be a quasi-pseudo-metric space. We say that a real function $f : S \to \mathbb{R}$ is semi-Lipschitz if there is a constant $K > 0$ such that for all $s, t \in S$,

$$\max\{\big(f(s) - f(t)\big), 0\} \leq Kq(s,t).$$

As far as we know, a systematic study of such operators was firstly done by Romaguera and Sanchis (2000). The following results can be easily proved using the same computation than in the original proofs of McShane and Whitney.

McShane extension for semi-Lipschitz maps. Let (S, q) be a quasi-metric space, a subspace (S_0, q) and a semi-Lipschitz function $f : S_0 \to \mathbb{R}$ with constant $K > 0$. Then the formula

$$f^M(s) = \sup_{t \in S_0} \{f(t) - Kq(t,s)\}, \quad s \in S,$$

provides a semi-Lipschitz extension with the same constant K.

A c-semi-Lipschitz real function $f : S \to \mathbb{R}$ is a map that satisfies the inequalities

$$\max\{\big(f(s) - f(t)\big), 0\} \le Kq(t,s), \quad s,t \in S.$$

We use this name because such a map is a semi-Lipschitz map in the conjugate space (S, d^{-1})

Whitney extension for c-semi-Lipschitz maps. Let (S, q) be a quasi-metric space, a subspace (S_0, q) and a c-semi-Lipschitz function $f : S_0 \to \mathbb{R}$ with constant $K > 0$. Then

$$f^W(s) = \inf_{t \in S_0} \{f(t) + Kq(t,s)\}, \quad s \in S,$$

is a c-semi-Lipschitz extension with the same constant K.

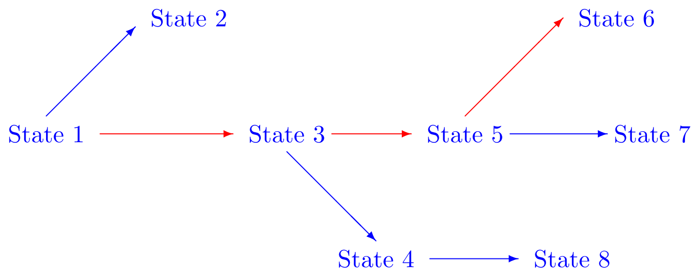
Finally, let us define the basic concepts regarding the graph structure that will be needed. All the definitions that we use can be found in the introductory sections of the book (Bondy and Murty 1976). The main graph type that will be considered here is what is called a polytree, which is a directed acyclic graph whose related undirected graph is a tree. Recall that a tree is an undirected graph that is connected, but has no cycles. Note that it is assumed that the graph is connected, but becomes unconnected if any edge is removed. Also, any couple of vertices in a tree can be connected by a unique simple path.

Although some other type of graph could be considered, having such an structure is necessary for the definition of what we call the bifurcation metric to make sense. It should be noted that this (quasi)-metric is not defined on the original graph, but on the space of all paths starting from the root (or a suitable subset of these paths) that can be defined on it, and which itself has again a graph structure. Indeed, the set of all these paths has itself a directed tree structure, that is defined in the obvious way (an arrow is defined from a path to any other path that coincides with it but has a new step).

We will clarify this in the next section. In order to do this, we need to define the following concept: given two paths in the graph s and t, the minimum path is the element $s \wedge t$ given by the common part of the branch in the directed tree that belongs to both of them, starting from the same root. In case we extend the notion to disjoint unions of polytrees (called forests), if there is no common part to s and t, we define $s \wedge t$ to be the empty path. This will be explained formally in Sect. 3 below.

3 Quasi-metric spaces of strategies and reinforcement learning

As we said in the Introduction, our arguments are based on the assumption of an underlying graph structure in the original space of the states of the system, which allows us to consider an adequate set of strategies—sequences of states—on it. Thus, each "real world" strategy is represented by a walk through the graph, as in the scheme below (in red).



In order to design our AI model of dynamical system based on topological graphs, we begin by introducing in this section a quasi-pseudo-metric structure for spaces of strategies. In a second step, we will implement the reward function, which will be the main prediction tool of the model. Note that the space of the states has a natural graph structure provided by the time evolution. Therefore, the fact that we consider quasi-metrics together with metrics is central. The intrinsic asymmetry that underlies the definition of a quasi-metric can be used to represent in the model the fact that, in general, time-dependent processes are not reversible. Using quasi-metrics we can model this fact: while the quasi-distance from a previous step to the next one can be a positive real number, the opposite step—representing going back in time—can be modeled to be forbidden by giving the quasi-metric a value equal to 0 or ∞, depending on the particular model.

3.1 Quasi-pseudo-metric spaces of strategies

Let $\mathcal{U}$ be a graph representing the states of a system and an associate space of strategies $\mathcal{S}$, that is the space of finite sequences (vectors with a finite set of coordinates) of states starting from the root of the tree endowed with a (quasi-pseudo)-metric q. The underlying graph is assumed to be a directed multiply connected graph with no cycles. Each step in any sequence of $\mathcal{S}$ represents a change from a state in $\mathcal{U}$ to another state in $\mathcal{U}$. We call "actions" to a change from a state to another.

It could be assumed that the graph $\mathcal{U}$ is also endowed with a quasi-pseudo-metric p; in this case, the properties of p could give some clues for a proper definition of the quasi-pseudo-metric q, using some duality relations. In this sense, Definition 1 in Zhang et al. (2019) provides a specific class of graph quasi-metric for $\mathcal{U}$ that could be used for this purpose.

In general, for defining such a quasi-pseudo-metric q we can consider several functions. In this section we will fix two of them. As we will see, together they represent the basic properties of the metric notions that are needed to model the (discrete) dynamical systems in which we are centering our study. These definitions are, as far as we know, unknown, and can be successfully adapted to the ML context we are interested in studying, together with the framework for implementing quasi-metrics to general RL schemes.

For the first one, we consider the *discrete metric* d in $\mathcal{U} \cup \{0\}$, where 0 is a new symbol not belonging to $\mathcal{U}$. It allows to represent finite sequences by embedding them in $(\mathcal{U} \cup \{0\})^{\mathbb{N}}$ as $s = (s_1, \dots, s_n) \mapsto (s_1, \dots, s_n, 0, 0, 0, \ldots)$. Thus, if $a, b \in \mathcal{U} \cup \{0\}$ we have that

$$d(a, b) = 1 \quad \textit{if} \quad a \neq b \quad \textit{and} \quad d(a, a) = 0.$$

Then we can define a function ρ in $\mathcal{S}$ as follows. If $s, v \in \mathcal{S}$,

$$\rho_W(s,v) = \sum_{i=1}^{\infty} w_i d(s_i, v_i)$$

where s_i denotes the i-th coordinate of s, v_i denotes the i-th coordinate of v, w_i are non-negative real numbers representing suitable weights and $W = \{w_i \,:\, i \in \mathbb{N}^+ \cup \{0\}\}$.

Lemma 1 *The function* $\rho_W : \mathcal{S} \times \mathcal{S} \to \mathbb{R}^+$ *is symmetric and satisfies*

$$\rho_W(s,v) \leq \rho_W(s,t) + \rho_W(t,v), \quad s,t,v \in \mathcal{S}.$$

Proof A direct computation proves the result; indeed, if $s = (s_1, \dots, s_n)$, $v = (v_1, \dots, v_m)$ and $t = (t_1, \dots, t_k)$ and we identify these elements with sequences in $\mathbb{R}$ as $(s_1, \dots, s_n, 0, 0, 0, ...)$, $(v_1, \dots, v_m, 0, 0, 0, ...)$ and $(t_1, \dots, t_k, 0, 0, 0, ...)$, we have that the sequences are eventually equal to 0. Thus, the sum conains only a finite set of terms, and so we can use the triangle inequality for the discrete metric d to obtain

$$\rho_W(s,v) = \sum_{i=1}^{\infty} w_i d(s_i, v_i) \leq \sum_{i=1}^{\infty} w_i d(s_i, t_i) + \sum_{i=1}^{\infty} w_i d(t_i, v_i) = \rho_W(s,t) + \rho_W(t,v).$$

Finally, note that it is clear by the definition, the fact that the sums have only a finite set of terms and the symmetry of d that $\rho_W(s,v) = \rho_W(v,s)$. □

However, it seems more convenient for the design of models for machine learning to consider the set of strategies based on the set of actions instead of the set of states of a system. That is, an strategy is a finite sequence of actions. In this case, the "bifurcation quasi-metric" would be more convenient. Let us explain this distance. The set $\mathcal{S}$ is defined by all the finite sequences of actions. It is based on the idea that a sequence of actions must represent a dynamics in a system. In fact, the elements of $\mathcal{S}$ can be understood as paths in a connected graph. A variation in an action of a sequence at a given point changes completely the states involved in the rest of the sequence, due to the nature of the involved quasi-metrics and of the underlying graph itself. That is, two sequences s_1 and s_2 that coincides in the first three actions produce up to this point exactly the same state of the system. However, a change in the fourth position changes the state produced in the system, at this moment and until the end of the sequence, even if the rest of the coordinates in both sequences—from the fifth on,—are the same. In the implementation of the quasi-norms we will use some stochastic elements to produce several possible states, from which we will choose the best one using a quasi-metric based reward function.

This motivates the definition of the following metric, that we denote by ρ_0. We call it the bifurcation metric. Consider the space of all finite sequences of actions with the following lattice operation: if $s = (s_1, s_2, \dots, s_m)$ and $t = (t_1, t_2, \dots, t_r)$, we define $s \wedge t$ as the sequence given by

$$s \wedge t = (s_1, s_2, ...s_n),$$

where n is the maximum value for which $s_1 = t_1, s_2 = t_2, ... \, s_n = t_n$. In case that there is no concidence in the first coordinate, we write $s \wedge t = \emptyset$.

Then we define

$$\rho_0(s,v) = \max\{length(s),\, length(v)\} \,-\, length(v \wedge s),$$

where *length*(z) is the number of nontrivial coordinates of the sequence of actions z, and $z \wedge h$ is defined as above. If $s \wedge v = \emptyset$ we put $length(s \wedge v) = length(\emptyset) = 0$.

Since we are interested in the construction of the most general analytic tool for including asymmetric effects, we will show in what follows that this metric can be defined as the maximum of a meaningful quasi-metric q and its conjugate q^{-1}.

We can define now a bifurcation quasi-metric q as follows. If $s, t \in \mathcal{S}$, we define.

$$q(s,t) = length(t) - length(s \wedge t);$$

in the same way, we can consider the conjugate quasi-metric, given by

$$q^{-1}(s,t) = q(t,s) = length(s) - length(s \wedge t).$$

Note that, if t is defined as the first n coordinates of s, then $q(s,t) = 0$; in fact this is the only case for which q equals 0, as can be easily seen. Note also that, as a consequence of this fact,

$$q^s(s,t) = \max\{q(s,t), q^{-1}(s,t)\} = \max\{q(s,t), q(t,s)\} = \rho_0(s,t) = 0$$

if and only if s and v coincide.

Remark 1 As we said before, the definition of these quasi-metrics is done with the aim of introducing into the model the asymmetry imposed by the fact that evolution over time is not generally a reversible process. That is why we work under the assumption that the graph is a tree. A process that advances with time cannot return to a previous state, so the quasi-norm takes a positive real value when time advances, but it could be infinite, or zero (this depends on the concrete case) if the opposite step is considered and the algorithm tries to return. Also, the metric of the bifurcation does not make sense if the graph contains cycles, although we believe that the definitions could be adjusted in some cases.

Let us next show that q is indeed a quasi-metric, and therefore $q^s = \rho_0$ is a metric as a direct consequence. It must be said that, although easy to prove, this lemma is central to the development of our results; the reason is that the McShane and Whitney formulas, necessary for the key step of extending semi-Lipschitz functions while preserving their constant, are no longer true if these inequalities fail. This assertion can be easily checked in examples.

Lemma 2 *Consider a set of strategies $\mathcal{S}$ defined by a set of actions. Then for every $r, s, t \in \mathcal{S}$, we have that*

$$q(r,t) \leq q(r,s) + q(s,t).$$

Consequently, and taking into account that the formula $\max\{q(s,t), q(t,s)\}$ *defines a metric, we have that q is a quasi-metric.*

Proof By definition, we clearly have that

$$length(r \wedge t) \geq \min\{length(r \wedge s), length(s \wedge t)\}.$$

Thus we obtain that

$$length(r \wedge s) + length(s \wedge t) \leq length(r \wedge t) + length(s).$$

Therefore

$$\begin{aligned} q(r,t) &= length(t) - length(r \wedge t) \\ \leq length(s) - length(r \wedge s) + length(t) - length(s \wedge t) &= q(r,s) + q(s,t). \end{aligned}$$

□

The next result summarizes the description and main properties of the set of quasi-pseudo-metrics that will be considered in the present paper.

Proposition 1 *Let $\mathcal{U}$ be a graph satisfying the requirements explained before. Let $\mathcal{S}$ be the associated set of strategies—sequences of actions, paths in the actions graph. Then*

1. *ρ_W is a pseudo-metric that becomes a metric if $w_i > 0$ for every $i \in \mathbb{N}$.*
2. *q and q^{-1} are quasi-metrics, and q^s is a metric.*
3. *For every $\alpha_\rho, \alpha_s, \beta_q, \beta_{q^{-1}}$ non-negative real numbers that sums 1, we have that the formula*

$$\tau(s,v) = \alpha_\rho\, \rho_W(s,v) + \alpha_s\, q^s(s,v) + \beta_q\, q(s,v) + \beta_{q^{-1}}\, q^{-1}(s,v), \quad s, v \in \mathcal{S},$$

defines a quasi-pseudo-metric. Moreover, it is a metric if $\alpha_\rho \geq 0$, $\alpha_s > 0$ and $\beta_q = 0 = \beta_{q^{-1}}$.

Proof 1. is a consequence of Lemma 1 and a simple argument for proving by contradiction that, if any w_i is zero, then we can find two different elements s, v of $\mathcal{S}$ such that $\rho_W(s,v) = 0$ (just take $s = (s_1, \dots, s_{i-1}, 0, 0, 0, ...)$ and $v = (s_1, \dots, s_{i-1}, w, 0, 0, ...)$ for $w \neq 0$).

The statements in 2. are just consequences of Lemma 2 and the other arguments above. Finally, 3. is obvious, just taking into account 1. and 2. □

3.2 Lipschitz maps in spaces of strategies

Consider now a finite subset $\mathcal{S}_0 \subseteq \mathcal{S}$. It represents the strategies that have been already checked, for which we already have an evaluation. We can consider now an evaluation map—a reward function—, that is a real function $f : \mathcal{S}_0 \to \mathbb{R}$ which, as we said, is supposed to be known. We consider it as a Lipschitz map. Since $\mathcal{S}_0$ is finite, the associated Lipschitz constant $K(f)$ is always finite.

We can always obtain a Lipschitz extension $\hat{f}$ of the evaluation function f to the whole space preserving the Lipschitz constant by using a McShane-type extension for Lipschitz functions. For example, we can use a convex combination of the McShane and Whitney formulas. Thus, we extend the evaluation function to the whole space of strategies, understood as sequences of possible actions. It can be already used to evaluate any strategy of the set, and so it provides a method for generating experience "to feed the system" from completely new situations that have not been checked in the real world. The options to be chosen will be given by a random election process, from which these new cases are obtained. This is the main tool of our purpose of reinforced learning, which have been already used in the context of the financial markets (Calabuig et al. 2020). Note that we can use either the metric ρ_W or the quasi metrics q and q^{-1} for extending the reward function f, since we know the corresponding extension theorems for all these cases (see Sect. 2).

The universe of states and the actions can be represented together as a directed graph as explained in Sect. 3.1. Often this representation allows a clear picture of the problem to be drawn (Driessens et al. 2006, Sect. 6.1.1). We will use such representation through the paper, since it also facilitates the use of some graph-based analytic programs that could be used for the design of more complete algorithms, as Neo4j or Gephi.

Remark 1 In Proposition 1 we have set the starting point to build the topological context of our models. However, if we are interested in the use of these ideas into a distance learning framework, this initial formula has to be modified in a recursive process, once it has been fixed in the definition of a metric τ with the corresponding weight in each term. We can use the extension of the reward function for this aim. Indeed, using the extension of the Lipschitz function $\hat{f}$ we can get a quasi-pseudo-metric $q_{\hat{f}}$ depending exclusively on the values of this function that will allow to define a new metric in the space. Thus, a formula like

$$p(s,v) = \alpha\, \tau(s,v) + (1-\alpha)\, q_{\hat{f}}(s,v), \quad s,v \in \mathcal{S}$$

for a coefficient $0 \le \alpha \le 1$, could give the next formula for a quasi-pseudo-metric defined in the space in the next recursive step. For example, a natural expression for $q_{\hat{f}}$ would be given by

$$q_{\hat{f}}(s,v) = \left|\hat{f}(s) - \hat{f}(v)\right|, \quad s,v \in \mathcal{S},$$

that can be improved by increasing the set $\mathcal{S}_0$ used to compute the extension $\hat{f}$.

4 First test examples: "The drunk man crossing the bridge" (DBProblem)

In this section we consider the extension of a particular reward function acting in a concrete metric space that models the following system. Consider a universe $\mathcal{U}$ with 3 possible situations at each state, and a set of 3 actions in them: "going ahead"—go ahead right = 1, go straight = 2, go ahead left = 3. The process begins with the "drunk man" at the beginning of the bridge and in the central position. Strategies are defined as finite sequences of actions. A strategy fails when a particular sequence of actions occurs in such a way that the "drunk man" falls off the bridge. For example, twice in a row to the right puts the drunk man in the water. The following chart (Table 1) represents a successful 5-step strategy. The process begins in the left central part of the table (asterisk), which represents the initial state 1 in step 0; after continuing to the left, the drunk man is still on the bridge after step 1 (state 2), and so on.

We use the bifurcation metric considered above. The evaluation function v is given in this case by the length of the strategy. It is supposed to represent the effectiveness of a strategy: the longer the drunk lasts on the bridge, the better the strategy. This of course can be computed for all the sequences, but we want to compare its values with the prediction of v provided by the Lipschitz extension of this evaluation function based in its values for a little number of strategies. The McShane formula will be considered together with the (symmetric) distance $\rho_0 = q^s$ —that is, $\alpha = 1$ and $\tau = \rho_0$ in the general metric model explained in Remark 1.

4.1 A first "Handmade" example

After some experiments rolling a die—that adds to the procedure an elementary (but necessary) probabilistic character,— we get the following sequences of actions. (The symbol 0/0 in the following computations is evaluated as 0.) In a well-structured algorithm, this first step should be performed following a typical Monte Carlo construction, after estimating a suitable probability distribution.

$$H_1 = (2,3,2,3), \quad H_2 = (1,1), \quad H_3 = (1,1),$$
$$H_4 = (2,3,3), \quad H_5 = (2,3,3), \quad H_6 = (1,1),$$
$$H_7 = (1,2,1), \quad H_8 = (3,3), \quad H_9 = (3,1,2,3,3,3).$$

Note that some of them are repeated. Following the notation of Sect. 3.2, we will consider the subset of strategies $\mathcal{S}_0 \subset \mathcal{S}$ defined by $\mathcal{S}_0 = \{H_1, H_2, H_3\}$. The idea is to develop a manageable example with a small set of states —just 3—, for which the evaluation function is known. Later we will compare the values of $\hat{v}$ —the Lipschitz extension of v— for the test set $\{H_4, H_5, H_6, H_7, H_8, H_9\}$ with the values of v for the elements of this test set, which can also be computed.

Indeed, a direct computation shows that

$$v(H_1) = 4,\ v(H_2) = 2,\ v(H_3) = 2,\ v(H_4) = 3,\ v(H_5) = 3,$$
$$v(H_6) = 2,\ v(H_7) = 3,\ v(H_8) = 2,\ v(H_9) = 6.$$

We will show now how we can get a Lipschitz estimate $\hat{v}$ for v using the values of v for the elements of $\mathcal{S}_0$. We will consider the McShane formula in this example.

(1) Since one of them is repeated, we get a pseudo-distance instead of a distance (that is in this case, symmetric) with representing distance matrix

$$D = \begin{bmatrix} 0 & 4 & 4 \\ 4 & 0 & 0 \\ 4 & 0 & 0 \end{bmatrix}.$$

Of course, this can be avoided just by removing repetitions in the experience. However, since we have obtained it as a result of an "experimental" process, we prefer to keep it as a different element: this is allowed by our pseudo-metric formalism.

(2) We compute the Lipschitz constant $K = \max_{i,k=1,2,3} \frac{|v(H_i)-v(H_k)|}{d(H_i,H_k)}$. We have that

Table 1 Scheme of an strategy on the bridge

State 1	State 2	State 3	State 4	State 5	State 6	State 7	State 8	⋯
	*							⋯
*		*						⋯
			*	*				⋯

$$\frac{|v(H_1)-v(H_2)|}{d(H_1,H_2)} = \frac{|4-2|}{4} = \frac{1}{2}, \quad \frac{|v(H_1)-v(H_3)|}{d(H_1,H_3)} = \frac{1}{2}, \quad \frac{|v(H_2)-v(H_3)|}{d(H_2,H_3)} = \frac{0}{0}.$$

Therefore, $K = 1/2$.

(3) Let us show the computation of the estimate for several $H_i, i > 3$.

- $\hat{v}(H_4) = \max_{k=1,2,3}\{v(H_k) - \frac{1}{2}d(H_k,H_4)\} = \max_{k=1,2,3}\{4-\frac{1}{2}2, 2-\frac{1}{2}3, 2-\frac{1}{2}3\} = 3.$
  Comparing with the value of v, we get $v(H_4) = 3 = \hat{v}(H_4)$. The predictive value of the McShane extension works in this case.
- $\hat{v}(H_6) = \max_{k=1,2,3}\{v(H_k) - \frac{1}{2}d(H_k,H_6)\} =\max_{k=1,2,3}\{4-\frac{1}{2}4, 2-\frac{1}{2}0, 2-\frac{1}{2}0\} = 2.$
  Obviously in this case $\hat{v}(H_6) = v(H_6)$, since in fact $H_6 = H_2 = H_3$.
- $\hat{v}(H_7) = \max\{2,1,1\} = 2$. In this case, we get $\hat{v}(H_7) = 2$, while $v(H_7) = 3$. Thus, the values are not coinciding, but they are similar yet.
- $\hat{v}(H_9) = \max\{4-\frac{1}{2}6, 2-\frac{1}{2}6, 2-\frac{1}{2}6\} = 1$, while $v(H_9) = 6$. The values in this case are quite different.

It can be seen that the method provides some reasonable predictions for the extension of the function v, although it might seem that we do not control to what extent we can trust the results obtained. However, note that the extension of v has to preserve the Lipschitz inequality for the constant $K = 1/2$, so that at least we have a uniform limit for the error.

4.2 An example involving quasi-metrics and semi-Lipschitz functions: the oracle function

We follow with the DBProblem but considering a new reward function and a genuine asymmetric structure to define the topology. In this case, the McShane and Whitney extensions give significantly different results, which is not surprising: we analyze them in two different sections below. We will use both q and q^{-1}—the bifurcation quasi-metrics,—and a new function that represents just the resulting state of an strategy: the oracle function, which takes the value 0 if the drunk falls into the river and one if he stays on the bridge.

4.2.1 The McShane semi-Lipschitz extension of the oracle function

Let us define the oracle function acting in the set of strategies as follows. We introduce a new character $\emptyset$. When it appears in a sequence, it means that the depicted strategy is over, as the drunk is already in the river.

If $s = (s_1, \dots, s_n)$ is a *complete strategy* —that is, a strategy that has finished when the drunk jumps into the river—, a sub-strategy is a sequence as $(s_1, \dots, s_r)$, for $r \le n$. Consider a set $\mathcal{S}_0 \subseteq \mathcal{S}$ generated by a set $S_{0,0}$ of complete strategies by considering all the sub-strategies of all the elements of $S_{0,0}$. In the model that supports this example, it is supposed that these strategies have been "experimentally" checked; of course, we compute them directly, since the algorithm for doing this is clearly defined. The oracle function is a map $\mathcal{O} : \mathcal{S}_0 \to \mathbb{R}$ defined as

$$\mathcal{O}(s) = 0 \ \textit{ if the next step/coordinate expected in s is } \emptyset$$

and $\mathcal{O}(s) = 1$ otherwise. As we said, this function is intended to represent the possibility of “survival" after the subsequent implementation of several actions: if after the application of a sequence of actions $s = (s_1, \dots, s_n)$, the system is still “alive" —that is, $(s_1, \dots, s_n)$ is not a complete strategy,—then the value is $\mathcal{O}(s) = 1$. Note that the same strategy can appear several times, according to our random way of generating the sample: in case of multiple values for a particular sub-strategy, we put the mean.

Let us compute now the semi-Lipschitz McShane extension $\hat{\mathcal{O}} = \mathcal{O}^M$ for a particular subset of complete actions of the set considered in Sect. 4.1. We chose in this case the set $\mathcal{S}_{0,0}$ defined by

$$\mathcal{S}_{0,0} := \{H_1 = (2,3,2,3),\ H_2 = (1,1),\ H_4 = (2,3,3)\},$$

whose elements represent complete strategies, to generate the test set $\mathcal{S}_0$, that is then defined as

$$\mathcal{S}_0 = \{(2),(2,3),(2,3,2),(2,3,2,3),(1),(1,1),(2,3,3)\}.$$

The computed values of $\mathcal{O}$ are:

$$\begin{aligned}&\mathcal{O}((2)) = 1, \mathcal{O}((2,3)) = 1, \mathcal{O}((2,3,2)) = 1, \mathcal{O}((2,3,2,3)) = 0,\\ &\mathcal{O}((1)) = 1, \mathcal{O}((1,1)) = 0, \mathcal{O}((2,3,3)) = 0.\end{aligned}$$

The semi-Lipschitz constant is the maximum of the next values.

$$\begin{aligned}&\frac{\mathcal{O}((2)) - \mathcal{O}((2,3,2,3))}{q((2),(2,3,2,3))} = \frac{1}{3}, \quad \frac{\max\{\mathcal{O}((2,3,2,3)) - \mathcal{O}((2)), 0\}}{q((2,3,2,3),(2))} = \frac{0}{0},\\ &\frac{\mathcal{O}((2,3)) - \mathcal{O}((2,3,2,3))}{q((2,3),(2,3,2,3))} = \frac{1}{2}, \quad \frac{\max\{\mathcal{O}((2,3,2,3)) - \mathcal{O}((2,3)), 0\}}{q((2,3,2,3),(2,3))} = \frac{0}{0},\\ &\frac{\mathcal{O}((2,3,2)) - \mathcal{O}((2,3,2,3))}{q((2,3,2),(2,3,2,3))} = \frac{1}{1} = 1, \quad \frac{\max\{\ \mathcal{O}((2,3,2,3)) - \mathcal{O}((2,3,2)), 0\}}{q((2,3,2,3),(2,3,2))} = \frac{0}{0}.\end{aligned}$$

It is easy to see that the semi-Lipschitz constant equals 1, and this is the case for every set $\mathcal{S}_0$ constructed as above. In our simulated experiments, we have found that the control of this constant is crucial for the competitiveness of the algorithm, and could be one of the reasons why it is inadequate or not recommended: if the constant increases rapidly, the results are progressively worse. Therefore, in this case the result is the semi-Lipschitz extension of $\mathcal{O}$ using the McShane formula, $\hat{\mathcal{O}}$, that is

$$\hat{\mathcal{O}}(H) = \max_{s \in \mathcal{S}_0}\{\mathcal{O}(s) - q(s,H)\}, \qquad \text{for every strategy } H.$$

An example: if we take $H = (2,1,1)$, we have that

$$\hat{\mathcal{O}}(H) = \max_{s \in \mathcal{S}_0}\{\mathcal{O}(s) - q(s,H)\} = \max\{1-2, 1-2, 1-3, 1-2, 0-3, 0-2, 0-3, 0\} = 0,$$

which gives the expected positive result, since this strategy is complete. In general, in this case it can be noted that by the definition of $\hat{\mathcal{O}}$, the only elements H such that $\hat{\mathcal{O}}(H) = 1$ are the ones that are (strictly) subsequences of a sequence in the original set $\mathcal{S}_{0,0}$, that is, the elements of $\mathcal{S}_0$ that are not complete actions.

4.2.2 The Whitney semi-Lipschitz extension of the oracle function

The formula that gives the extension in this case is,

$$\mathcal{O}^W(s) = \inf_{t \in S_0} \{f(t) + Kq(t,s)\}, \quad s \in S_0,$$

However, we cannot apply it; using the setting that we shown in the example of the previous section, we have that

$$\frac{\mathcal{O}((2)) - \mathcal{O}((2,3,2,3))}{q((2,3,2,3),(2))} = \frac{1-0}{1-1} = \frac{1}{0},$$

what means that the oracle function $\mathcal{O}$ cannot be considered as a c-semi-Lipschitz function. Following Romaguera and Sanchis (2000, p.294), we can say that the reason is that this function is not $\leq_{q^{-1}}$-increasing. Therefore, the Whitney semi-Lipschitz extension of $\mathcal{O}$ cannot be constructed in this case.

5 Semi-Lipschitz extensions as decision tools: examples of the forecasting method

In this section we show how the extension of Lipschitz-type functions behaves in order to predict the evolution of a system. To do so, we focus on the prediction step, that is, on the calculation of the extension based on the experience stored in the memory. In the next step, the iterative learning algorithm to build a typical machine learning procedure would consist in the storage of the information obtained in each step, that would increase the size of the metric space from which the extension can be computed. We use the example of the DBProblem as a benchmark. We will consider the oracle function $\mathcal{O}$ studied in Sect. 4 as reward function and a slight modification of its values including some probabilistic estimate. These functions can be computed for every strategy in the model, so we can compare the estimate that the model provides—what will be called the success function I,—with its exact value to test the effectiveness of the procedure. This function I will be computed as a Lipschitz extension of the oracle function when it is evaluated in a specific small training set $\mathcal{B}$. Concretely, it will be defined as the mean of the McShane extension and the Lipschitz extension of $\mathcal{O}$, originally defined in an eight-vectors-set $\mathcal{B}$.

Recall that we are considering the set of strategies defined as sequences of actions over the three-position states underlying the DBProblem. Over this space we consider the bifurcation metric defined in Sect. 3.1. Again, we take a small set of strategies in the set of actions for checking the model. Recall that 1 means that "the drunk is still on the bridge," (DonB) and 0 that "the drunk is already in the river" (DinR). We use an algorithm for checking some particular situations. As we said we consider a success function I—that plays the role of extension of the reward/oracle function studied in the previous section,—and gives an estimate of the expected success of an strategy. Our success function I is defined as the mean of the McShane and Whitney extensions of the oracle function provided in the previous section. This function takes the value 0 if DinR, 0.5 to indicate that the algorithm does not give any estimate for the given case, and 1 if DonB.

Let us show three specific computations to explain how the algorithm works. We do not try to prove fundamental properties of the proposed algorithms (tractability, scaling,...), we intend to show how the algorithm provides right answers even when it is supported by very small data sets.

Example 1 We start by considering a very small set of strategies. It is assumed that it comes from a direct experiment and that the system has stored only these cases at an early stage of the evolutionary process. Remember that, in this section, we are focusing on the analysis of the efficiency of the extension procedure, based on the following set of eight strategies together with the corresponding values of the oracle function. Our idea is to show that, even with such a small training set, the method provides a relevant rate of successful prospective results. Also relevant is the fact that the algorithm *does not fail*: if it cannot provide a reliable solution, it gives "indeterminate" as response.

Strategy	Oracle	Strategy	Oracle	Strategy	Oracle	Strategy	Oracle
(2,2,3,1)	1	(2,2,3,3)	0	(1,2,3)	1	(2,3,3)	0
(1,2,3,1,3,3)	0	(2,3,1)	1	(1,1)	0	(1,2,1)	1

To check this, we consider the set of all the 3-steps walks on the bridge as testing set, even if "the drunk man falls in the water" in the second step. This means that it is allowed to try to enter the bridge again on the third step, although in the game we assume that, once he has fallen, the walk cannot be considered successful.

As we said, we choose as extension I of the oracle function $\mathcal{O}$ the mean of the McShane and the Whitney extensions, that is,

$$I(s) = 1/2\big(McShane(s) + Whitney(s)\big), \quad \text{for each strategy } s.$$

The 3-steps walks are ordered as follows (by rows). Note that, as we said above, strategies that have already finished at the second step, as for example (1, 1, 2)—the drunk man is already in the water after two steps—, are considered to be evaluated in the third step. Of course, the evaluation has to be 0 in this case.

(1,1,1)	(1,1,2)	(1,1,3)	(1,2,1)	(1,2,2)	(1,2,3)	(1,3,1)	(1,3,2)	(1,3,3)
(2,1,1)	(2,1,2)	(2,1,3)	(2,2,1)	(2,2,2)	(2,2,3)	(2,3,1)	(2,3,2)	(2,3,3)
(3,1,1)	(3,1,2)	(3,1,3)	(3,2,1)	(3,2,2)	(3,2,3)	(3,3,1)	(3,3,2)	(3,3,3)

Next we show a comparison of I with $\mathcal{O}$ (whose exact value can be directly computed) to illustrate the power of the algorithm as a forecasting tool. In Fig. 1, the exact values of the function $\mathcal{O}$ are shown. The following Fig. 2 shows the success function I—that is, the extension of the oracle function— for these 21 strategies. Each point represents the value of the order position—first coordinate—and the corresponding extension of the oracle function—second coordinate—for each strategy.

Thus, the set of eight initial vectors allows to obtain an estimate for all the 3-steps strategies for which the algorithm decides. Since we define it as the mean of the McShane and the Whitney extensions, we get also values of 0.5 for the success function I, that represent in the model the 3-steps walks for which the algorithm cannot provide a reasonable estimate.

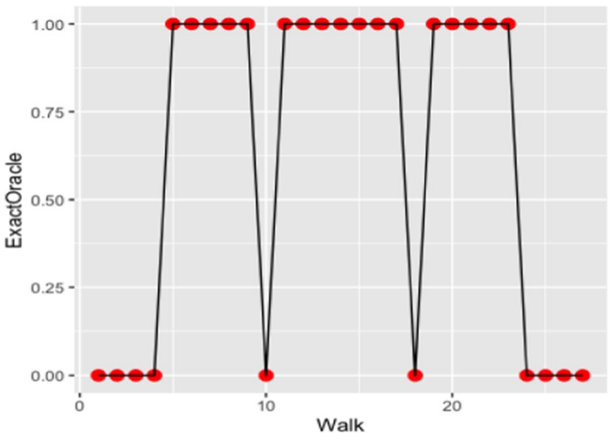


Fig. 1 Representation of the exact values of oracle function of Example 1

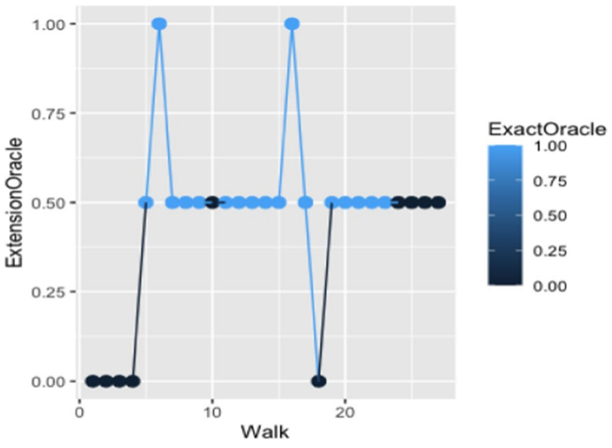


Fig. 2 Representation of the extension *I* of the oracle function—the estimate obtained by using our extension technique—of Example 1

For the rest of the cases, the algorithm gives also some information: obviously, it gives a right result when the 3-steps walk that is considered belongs to the original training set. But it also is right for other cases, the ones at the beginning of the list. It can be seen that the results are consistent with the "honesty" of the algorithm: it gives results for the case that we can trust these results, minimizing the probability of getting a wrong answer. Summarizing, we have obtained the following.

* Indeterminate Cases (the algorithm gives the information that it cannot provide any estimate): we get that only for 7 of 27 cases the algorithm gives an answer. That is, the rate of indetermination is 74.07%.
* Success Rate (for the cases for which the algorithm gives an answer): for the 7 cases for which the algorithm gives a prediction, we get a 100% of success; 4 of these cases belong to the original training set, so the algorithm gives a positive answer for 3 cases.
* The algorithm does not give wrong results. In cases where the information is not sufficient to give a correct answer, the algorithm gives the value 0.5, indicating that it is not able to find a right result.

Example 2 Let us now use all the 2-step strategies as training set $\mathcal{S}_0$. They can be seen in Table 2, together with the corresponding values of the oracle function. We consider the set ordered by rows. In this case, we change the definition of the oracle function, that is now defined by the uniform probability of staying on the bridge in the next step from the position defined by the given 2-step walk. As in the case of Example 1, forecasting is made also on walks that could be already out of the bridge.

With this training set and using the bifurcation (quasi-)metric, we analyze the prospective on the oracle function for all the sets of 3-steps and 4-steps.

Thus, the results for the case of 3-step walks, when the training set that we consider is the one given by all the 2-step walks, is

$$Relative\ Error = \frac{1}{21}\sum_{i=1}^{21}\left|\mathcal{O}(s_i) - I(s_i)\right| = 0.1975.$$

The quadratic error is $= \frac{1}{21}\sqrt{\sum_{i=1}^{21}\left|\mathcal{O}(s_i) - I(s_i)\right|^2} = 0.0605.$

Concretely, the values of the reward—the exact oracle function—can be found in Tables 3 and 4. In the first one we find the exact values of the reward function, where we have considered the order by rows, while the estimate of this function provided by the algorithm are presented in the next Table 4. The comparison of these results are presented in Fig. 3.

It can be seen that there are 8 cases for which the algorithm gives the value 2/3 while the correct one is 1, and there are just 4 cases for which the algorithm gives 2/3 and the exact value is 0. For the resting 9 cases the algorithm gives the right result.

Let us show now the results for the all the 4-step walks. They can be seen in Fig. 4. In this case, we have the same relative error of 0.1975, while the relative quadratic error is 0.0349. The reader can note that the forecasts provided for these cases are similar to the cases of 3-steps and 4-steps walks, and the undefined cases—cases for which we only have the probabilistic estimate—do not disappear, but neither do they increase disproportionately. As can be seen in Fig. 5, even for these cases the algorithm provides useful information.

Of course, this is not in general the case. In real world applications, we will find some known situations from which we will extend the model to all the possible cases. In the context of distance learning, this will be done improving the topology of the graph by adding some experimental information as explained in Sect. 3.2. In the next section we will explain a typical context for applying this methodology. A predictive tool to compute the behavior of a financial market is presented, with the aim of giving a systematic way of improving the original distance given for the space, that is constructed by using the bifurcation metric.

Table 2 Set of strategies in the training set $\mathcal{S}_0$

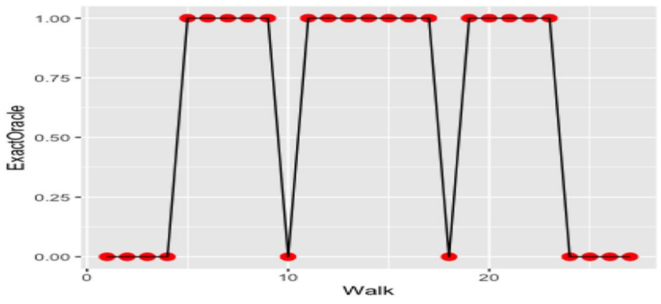
Strategy	Oracle	Strategy	Oracle	Strategy	Oracle
(1,1)	0	(1,2)	2/3	(1,3)	1
(2,1)	2/3	(2,2)	1	(2,3)	2/3
(3,1)	1	(3,2)	2/3	(3,3)	0

Table 3 Exact values of the oracle function for 3-step walks

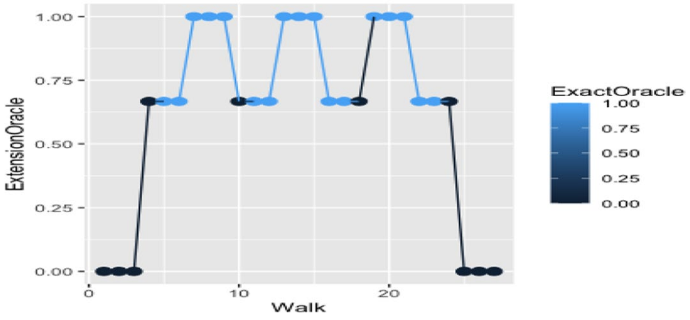
0	0	0	0	1	1	1	1	1
0	1	1	1	1	1	1	1	0
1	1	1	1	1	0	0	0	0

Table 4 Estimate *I* of the oracle function for 3-step walks (success function)

0	0	0	2/3	2/3	2/3	1	1	1
2/3	2/3	2/3	1	1	1	2/3	2/3	2/3
1	1	1	2/3	2/3	2/3	0	0	0



(a) Exact values of the oracle function.

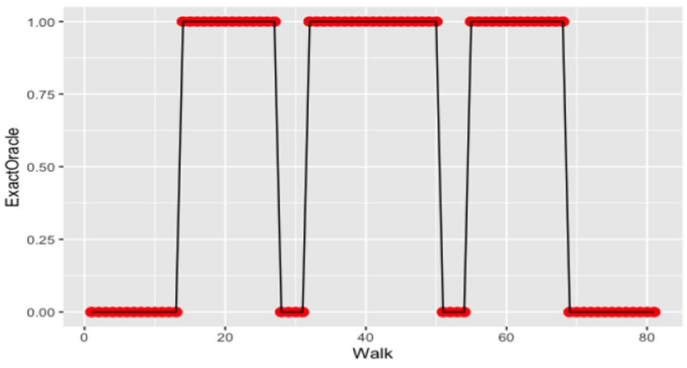


(b) Extension of the oracle function.

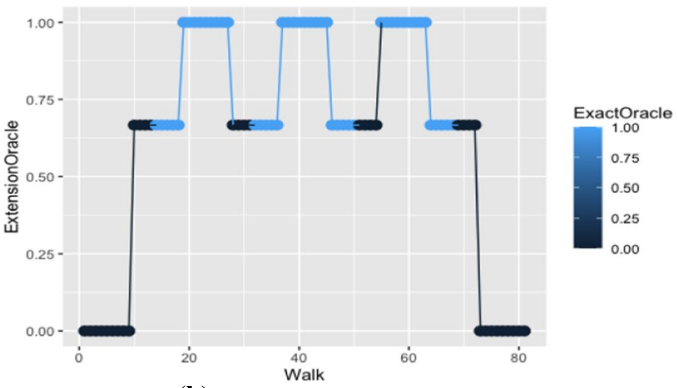
Fig. 3 Comparison of the oracle function and its extension for all the 3-step walks

6 Asymmetric reward functions: a computational example

The use of metrics defined by the symmetrization of an asymmetric distance, as an essential step of a reinforcement learning algorithm, has been shown in the academic example presented in the previous section. Given a closed set of "experiences" in the context of the DBProblem, we have shown how the system can create an evaluation tool to choose the next step in the algorithm , using Lipschitz extensions of adapted indices. However, this example does not show all the main properties that the proposed methodology can offer. First, the non-symmetric ideas underlying the definitions involved do not appear explicitly



(a) Exact values of the oracle function.



(b) Extension of the oracle function.

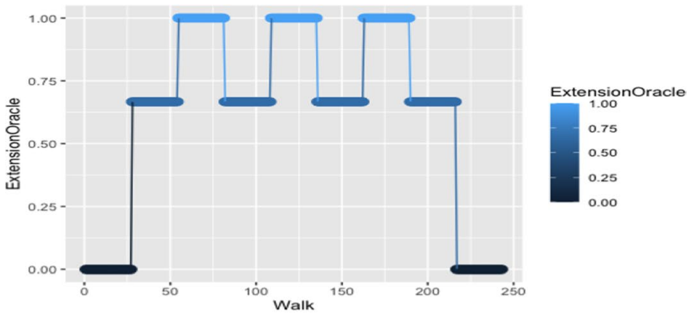
Fig. 4 Comparison of the oracle function and its extension for all the 4-step walks

in the formulas, although we have used them for the design of the tool. Secondly, a reinforcement learning procedure has to include the recursion tool to show the progress of the algorithm.

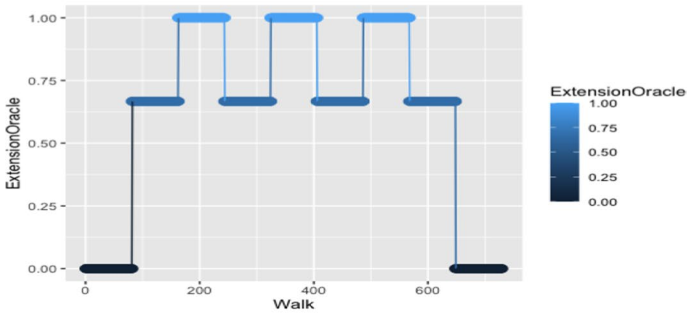
The fact that the set of actions considered in the systems under study define a tree (a *directed* graph), suggests that the natural tools to face it must be asymmetric. In fact, we will show in this section that for the complete construction of a reinforcement learning algorithm, it is more convenient that the associated metric notions—as well as the McShane-Whitney extension formulas that appear,—are the non-symmetric versions of the bifurcation metric.

The problem we present now can be understood as a general version of the DBProblem. Note that it can be made as large as we want, so it is an easy benchmark to check to what extent the algorithm could be efficient. This matter will be considered at the end of this section. There is a “field"—a rectangular grid,—of length N full of hidden holes, and the device has to cross it with the help of an automatic system. At each step, the device can advance left, forward or right, and for that it has to choose which—among a succession of randomly generated paths,—it considers as a possibility to advance on the board, and pays a fee to check its success.

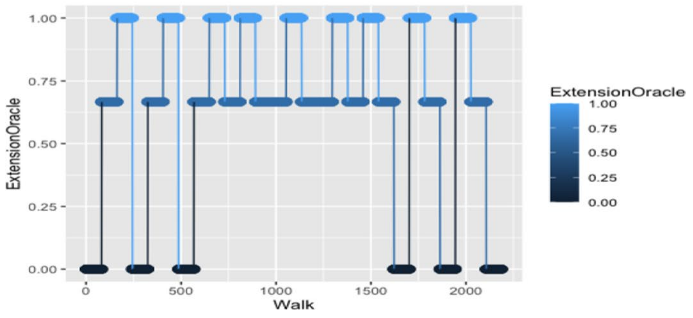
The algorithm follows by successive steps with the following rule:



(a) Extension of the oracle function for 5-step walks.



(b) Extension of the oracle function for 6-step walks.



(c) Extension of the oracle function for 7-step walks.

Fig. 5 Extension of the oracle function for longer walks (5, 6 and 7-step walks)

(1) The agent "sends" D times a device that follows random walks composed by n steps ($1 \leq n \leq N$), and gets a final score E for all of them: $E(s) = 0$ in case the device has found a hole, or $E(s) = 1$ in case the robot is still "on the board". For simplicity, we assume that there is a natural number R such that $N = R \cdot n$.

(2) These steps are registered in the memory of the system: $D_0(1)$ is the set of walks with score equal to 1 and $D_0(0)$ the walks with score equal to 0.

(3) The agent chooses randomly again a set A of D walks composed by $2 \cdot n$ steps. The bifurcation quasi-metrics are now considered: for every $t \in A$, we compute

$$q(s,t) = length(t) - length(t \wedge s), \quad s \in D_0(1), \quad \text{and} \quad q^{-1}(s,t) = q(t,s).$$

Note that the meaning of the evaluation of these quasi-metrics concerns the comparison of the trajectories involved: $q(s, t)$ represents the similarity of the proposed new walks s that go beyond, with the reference set of successful paths $t \in D_0(1)$, while $q^{-1}(s,t)$ gives how similar the new paths are to the unsuccessful ones $t \in D_0(0)$. A simple computation (similar to the ones done in the previous sections) gives that the Lipschitz constant K associated to the inequality $E(s) - E(t) \leq K \cdot q(s,t)$ is equal to 1. This justifies the definition of the next reward tool.

(4) We use the quasi-metric q^{-1} to exclude wrong paths of the set A : if $q^{-1}(s,t) = 0$ for any $t \in D_0(0)$, then s cannot provide a successful path, so we will only consider the set $A_0 \subset A$ given by

$$A_0 := \{t \in A : q^{-1}(s,t) \neq 0, \, t \in D_0(0)\} = \{t \in A : q(t,s) \neq 0, \, s \in D_0(0)\}.$$

In particular, for all the elements $t \in A_0$ we have that $E(t) = 1$. Now, we use the non-symmetric McShane extension formula to evaluate the better options among the paths of the set A_0 :

$$E^M(t) := \max_{s \in A_0} \{E(s) - K \cdot q(s,t)\} = 1 - \min_{s \in A_0} q(s,t).$$

We choose the subset $A_1 \subseteq A_0$ of the walks for which the values of E^M attain the maximum value. We evaluate them and choose the new sets $D_1(1) \subseteq A_1$—successful walks,—and $D_1(0) \subseteq A_1$—unsuccessful walks.

(5) We repeat the process from Step (2) on by choosing $D_0(1) \cup D_1(1)$ and $D_0(0) \cup D_1(0)$ to play the roles of $D_0(1)$ and $D_0(0)$. If there is not a "wall of holes" for the set of all the involved trajectories, the algorithm stops with a right complete path after R iterations.

It should be noted that the cost of the procedure is focused on the experimental verification of the success of a certain path, rather than on the calculation of the McShane extension for a new path obtained at random: remember that the board is unknown. Thus, the algorithm has to store also the success rate of the randomly generated paths as a function of the McShane extension, in order to choose the best distribution of the randomly generated subset to check at each step. This distribution provides information about how the "local" fill rate of the board (here "local" means "around the successful paths" tested in the previous step), and is the cornerstone for improving the efficiency of the algorithm. This point will be discussed in some detail later.

The given algorithm provides an example of the use of non-symmetric metric instruments. Note that this is only an example, as the potential use of the conceptual platform explained is broad and not limited to this topic. We have considered this algorithm to show that these tools allow to give a reinforcement-learning-type method to find a safe path through the network. Recall that in reinforcement learning, the learning agent interacts with an initially unknown environment and modifies its action policies to maximize its cumulative payoffs. Thus, RL provides an efficient framework to solve learning control problems which are difficult or even impossible for supervised learning and traditional dynamic programming methods In a sense, in our case the sequential aspect is not explicitly given by a value function defined as the sum of sequential contributions, but by a kind of "sum of sets", in the sense that learning occurs by including new elements in the set of known situations. We can design an artifact so that when an action is successfully performed, the sequence of steps leading to it is included in the set of successful experiences, and then the quasi-pseudo-metric compares the proposed new options (which has been provided by a random procedure) with the experience, which is growing step by step. Each new action of the agent is a benefit for the knowledge that has been attained about a given environment in order to maximize this "accumulated reward". This is how increasing knowledge is acquired. If the basic comparison set is left unchanged (as in the first examples written in previous sections), learning is reduced. However, if we increase this set step by step we get the cumulative effect that characterize reinforcement learning; actually, a value function (playing the role of cumulative reward) could be obtained by adding particular evaluations of the quasi-metric at each step.

Below are the results of a small example, in which $N = 12, n = 4$, and therefore $R = 3$. The grid has been randomly generated with a filling factor of 30% of holes. So, the distribution of the holes follows the following rule: at each step of one unit, the probability $P(1)$ of a successful result is $P(1) = 1 - 0.3 = 0.7$. Thus, the probability $P(n)$ of a random path of size n , not falling into a hole, is $P(n) = (0.7)^n$.

The application of the algorithm is explained below. In Fig. 6, a representation of the solution obtained to the problem is given. Figures 7 and 8 show different steps in the evolution of the algorithm, in which the path obtained in the corresponding step is shown in red, and in green some paths obtained randomly to be tested to obtain the new successful path.

In Fig. 7, 8-steps optional paths randomly obtained at the first step are shown. The algorithm chooses the ones that are closer in the quasi-norm to the first 4-steps path that is shown to be successful. At this point, only the first four steps of the successful (red) path are known.

Figure 9 shows a set of paths p which satisfy that the asymmetric distance $q^{-1}(p_0, p) = q(p, p_0)$ to a wrong path p_0 is equal to 0 (that is, a path whose qualification is equal to 0. i.e., the device has detected a hole). Following the proposed algorithm, this means that these paths have to be excluded from the set of potential suitable solutions chosen at random.

Let us now analyze some elements concerning the suitability of the proposed method. Note that performing a solution to such a problem using other RL algorithms is beyond the scope of the present work, due to the different nature of the standard candidates that could be used. The natural ones would be those related to Markov Decision Processes. To fully test the efficiency and compare our method with a standard algorithm it would be necessary to first design a competing algorithm for our method. The aim of this paper is not to show this, but to provide a different topological setting for understanding RL on discrete dynamical systems.

However, we can show some hints in the direction of improving the efficiency of the algorithm. Let us now consider the procedure for choosing the distribution of paths to be checked as a function of the corresponding values of the McShane extension. Fix a path p_0 of size N. The probability—given a path p of size N—,to get that the quasi distance $q(p, p_0) = k$ for $0 \leq k \leq N$, is $Prob = (1/3)^k(2/3)$. On the other hand, the q-McShane extension of the index for any path p of length N using the set $\{p_0\}$ for a successful path p_0, is

$$McS(p) := E^M(p) = \sup_{g \in \{p_0\}} \Big(E(g) - d(p, g) \Big) = 1 - d(p, p_0).$$

Therefore, given a set of M randomly chosen paths of size N, we have that the following distribution of paths having an index $q - McS(p) = s, 1 - N \leq s \leq 1$, is

$$M\,(1/3)^k\,(2/3), \quad \text{for} \quad 1 - (N - k) = s,$$

that is, $M\,(1/3)^{(N+s-1)}\,(2/3)$. In Fig. 10 a representation of the random distribution of successful paths is given for $M = 10000$.

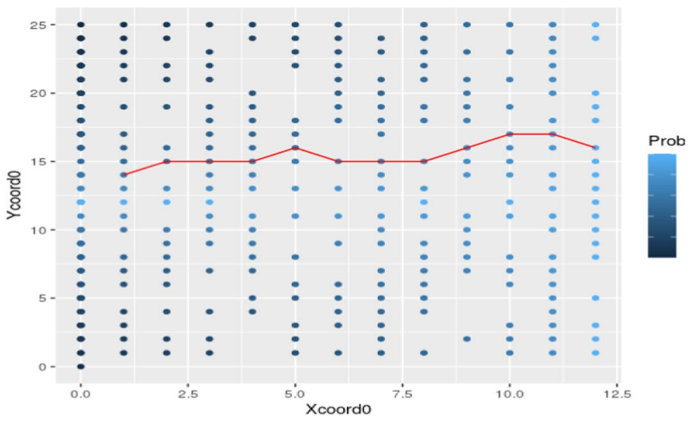


Fig. 6 Representation of the final path provided by the algorithm. The intensity of the blue color of the points indicates its probability of being crossed by a random path (the ligther the color, the greater the probability)

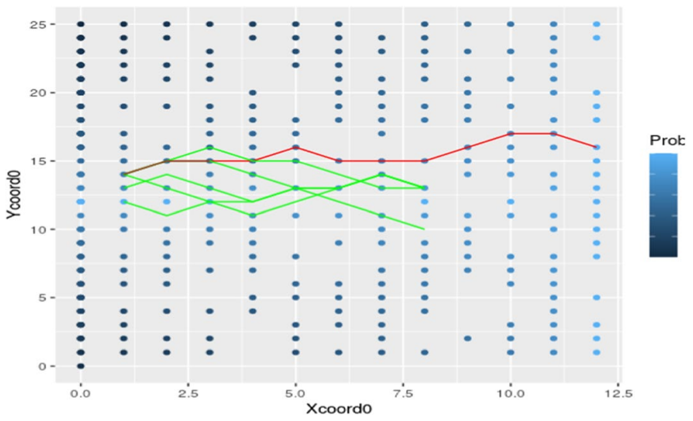


Fig. 7 Representation of the 8-steps optional paths randomly obtained at the first step

However, to improve the efficiency of the algorithm, this distribution can be changed to use one that can help find alternative ways to get a path. The motivation is that, depending on the (local) accumulation of holes (filling factor), it might be better to use a distribution that more closely matches the successful paths. If there are no holes, the most efficient strategy is to take for step n only paths that match one that is known to be successful in step $n-1$. By following this rule, the algorithm could skip the "walls of holes" that could occasionally appear after a previous successful path. Note that, for this, the complete history of the process is needed, and all previous steps must be recorded. This is a considerable difference with the setup of Markov-type algorithms.

To do this, we can estimate the filling ratio of the board only using the distribution calculated above. As we have explained, the probability that a path of length n is successful, is $P = r^n$. Thus, after checking a first set of random walks, we can get an estimate of P as

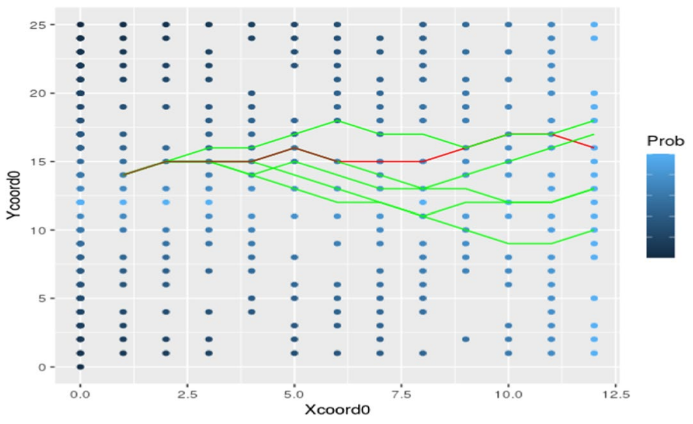


Fig. 8 Representation of the 12-steps optional paths randomly obtained at the second step. Only the first eigth steps of the successful (red) path are known

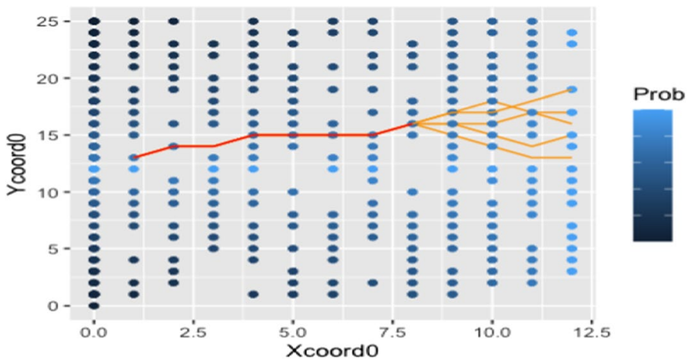


Fig. 9 Representation of five invalid random paths associated to a non succesfull stored path (increased dot size for better visualization of the holes)

$$EstP = \frac{\text{number of successful paths}}{\text{total number of paths checked}} \approx r^n,$$

and so $r \approx (EstP)^{1/n}$. The estimate of r can be improved at each step by adding to the total number of paths considered the new ones. Using this value, we could find a more adequate distribution of the frequency with which we accept as candidates possible paths with different McShane extensions, than the natural distribution obtained by random choice. Recall that R represent the increment of single steps in the board that is taken at each step of the algorithm, and we are placed after the first successful step has been done. For example, if $r \approx 1$ the acceptance of new paths to try could be 90% with $McS(p) = 1 - R$ and 10% with $McS(p) = 1 - R - 1$. However, if $r \approx 0.5$, a distribution as 30% of paths with McS equal to $-R, -R - 1$, and $-R - 2$, and 10% with $McS(p) = 1 - R$, could be chosen.

Thus, this distribution can be selected by the analyst according to his/her experience; automatic ways of doing so could also be easily designed, using elementary probabilistic arguments. In addition, a more precise analysis could be done to improve efficiency,

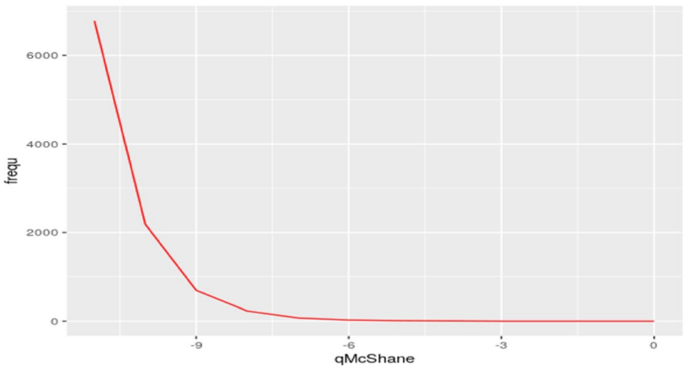


Fig. 10 Random distribution of paths having different values of the McShane non-symmetric extension with respect to a fixed 8-steps-path

considering that the distribution of holes does not have to be homogeneous throughout the board. In addition to the lengths of the previously used paths, the directionality of these paths could also be considered, which would allow choosing the direction on the board, rewarding the most successful ones.

To follow with the example, we have considered seven cases of boards with filling factors 0.5, 0.6, 0.7, 0.8, 0.9, 0.95 and 1, and in each one a successful path p of length 4 has been set. Figure 11 provides a representation for each filling factor of the frequencies of success of length-8-paths that have been randomly chosen depending on the value of the McShane extension with respect to p. Specifically, at each value of s, the dependent variable *Freq* shows the probability of success of a random path that has a McShane extension greater than or equal to s. Using the experimental estimate of this curve, the algorithm chooses as explained the best value of the McShane extension to optimize the solution, improving efficiency in this way. For example, the best option for the filling factor 0.9 is to choose random paths p with $E^M(p) = McS(p) \geq -5$.

Finally, although no specific calculations have been made, it should be noted that the scalability of the algorithm is expected to be good in terms of board size. The complexity of the system increases as the length of the required solution increases, but it can be observed that, by definition, the same computational scheme is repeated at each step, so only a linear increase in scalability could be expected. However, a full analysis of this issue will have to be carried out in future research steps on this topic. It is fundamental to the development of concrete algorithms that exploit graphs endowed with non-symmetric distances and extensions of semi-Lipschitz functions for the creation of new ML techniques.

7 Conclusions

A new conceptual framework—not based on vector representation—is introduced for the design of predictive tools for dynamic systems. The non-symmetry of the metric notions, which allows to emulate the non-reversibility of evolution over time, plays a crucial role. Combining (directed) trees of states with quasi-pseudometrics in the spaces of the strategies on those trees—finite sequences of actions,—we obtain graph/metric structures that allow us to define models for these systems.

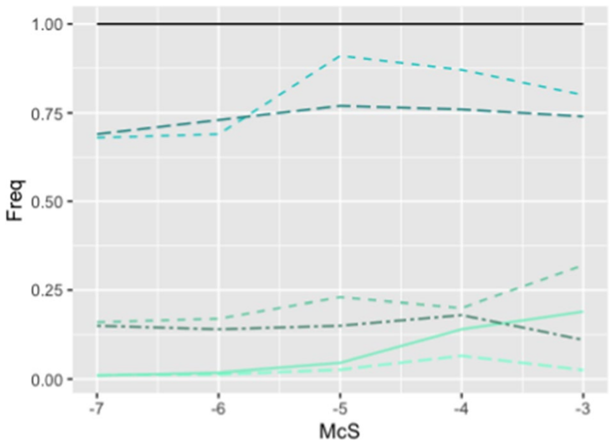


Fig. 11 Efficiency of the algorithm. Representation of the success frequencies as functions of the McShane extension for seven examples with different filling factors: 0.5, 0.6, 0.7, 0.8 in the lower part, and 0.9, 0.95 and 1 in the upper part, ordered by color intensity

The reward function is defined by a quasi-pseudo-metric over a training subset of strategies, that provides the instrument to determine which are the best. The main property of the proposed tools is that they combine graph structures together with asymmetric distances—quasi-metric,—in order to create a suitable framework for the design of AI algorithms for prediction in dynamic systems. In a second step, and using the well-known mathematical results about the extension of semi-Lipschitz functions in metric spaces, we extend the reward function that controls the Lipschitz constant, which guarantees the fundamental principle that states that similar strategies would have similar rewards.

We have checked some algorithms based on these ideas, considering some examples around the 'Drunk Man Crossing the Bridge" problem (that we called the DBProblem), and we have presented the corresponding computational results, together with some related explanations. In Sect. 6, we present an algorithm to solve a generalization of this problem, explaining a complete ML procedure designed for this purpose, along with some ideas for the analysis of some of its main properties (efficiency, scalability,...). The proposed method could be applied, for example, in the development and improvement of ML techniques designed for the financial markets, for which similar procedures based on the Lipschitz extensions have already been proposed (see Calabuig et al., 2020) and references therein).

Acknowledgements Both authors gratefully acknowledge the support of the association Tactical Whistleblowers, KPI, and Observatori de Transparència y Gestió de Dades, Generalitat Valenciana-UPV.

Author Contributions Both authors H. Falciani and E.A. Sánchez Pérez contributed equally to the work.

Funding Open Access funding provided thanks to the CRUE-CSIC agreement with Springer Nature. Support of Tactical Whistleblower, KPI, and Observatori de Transparència y Gestió de Dades, Generalitat Valenciana-UPV.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

Consent for publication Both authors consent the publication of the paper.

References

Asadi, K., Misra, D., & Littman, M.L. (2018). Lipschitz continuity in model-based reinforcement learning. arXiv preprint arXiv:180407193.

Barnes, J. A., & Harary, F. (1983). Graph theory in network analysis. *Social Networks, 5*(2), 235–244.

Barrett, C. L., Chen, W. Y., & Zheng, M. J. (2004). Discrete dynamical systems on graphs and boolean functions. *Mathematics and Computers in Simulation, 66*(6), 487–497.

Bellman, R. (1957). *Dynamic programming*. Princeton University Press.

Bellman, R. (1958). On a routing problem. *Quarterly of Applied Mathematics, 16*(1), 87–90.

Bondy, J. A., & Murty, U. S. R. (1976). *Graph theory with applications*. Macmillan.

Boutilier, C. (1999). Sequential optimality and coordination in multiagent systems. *IJCAI, 99,* 478–485.

Brandes, U. (2005). *Network analysis: methodological foundations* (Vol. 3418). Springer.

Bronstein, M. M., Bruna, J., LeCun, Y., Szlam, A., & Vandergheynst, P. (2017). Geometric deep learning: Going beyond Euclidean data. *IEEE Signal Processing Magazine, 34*(4), 18–42.

Buckley, F., & Harary, F. (1990). *Distance in graphs*. Addison-Wesley.

Bunke, H., & Shearer, K. (1998). A graph distance metric based on the maximal common subgraph. *Pattern Recognition Letters, 19*(3–4), 255–259.

Bu, C., Yan, B., Zhou, X., & Zhou, J. (2014). Resistance distance in subdivision-vertex join and subdivision-edge join of graphs. *Linear Algebra and its Applications, 485,* 454–462.

Calabuig, J. M., Falciani, H., & Sánchez-Pérez, E. A. (2020). Dreaming machine learning: Lipschitz extensions for reinforcement learning on financial markets. *Neurocomputing, 398,* 172–184.

Camacho-Collados, J., & Pilehvar, M. T. (2018). From word to sense embeddings: A survey on vector representations of meaning. *Journal of Artificial Intelligence Research, 63,* 743–788.

Cao, Q., Ying, Y., & Li, P. (2012). Distance metric learning revisited. *Joint European Conference on Machine Learning and Knowledge Discovery in Databases* (pp. 283–298). Springer.

Cao, S., Lu, W., & Xu, Q. (2016). Deep neural networks for learning graph representations. *AAAI, 16,* 1145–1152.

Casteigts, A., Flocchini, P., Quattrociocchi, W., & Santoro, N. (2011). Time-varying graphs and dynamic networks. *International Conference on Ad-Hoc Networks and Wireless* (pp. 346–359). Springer.

Chami, I., Abu-El-Haija, S., Perozzi, B., Ré, C., & Murphy, K. (2020). Machine learning on graphs: A model and comprehensive taxonomy. arXiv preprint arXiv:200503675.

Chávez, E., Navarro, G., Baeza-Yates, R., & Marroquín, J. L. (2001). Searching in metric spaces. *ACM Computing Surveys (CSUR), 33*(3), 273–321.

Chebotarev, P. (2011). A class of graph-geodetic distances generalizing the shortest-path and the resistance distances. *Discrete Applied Mathematics, 159*(5), 295–302.

Chen, H., & Giménez, O. (2010). Causal graphs and structurally restricted planning. *Journal of Computer and System Sciences, 76*(7), 579–592.

Chen, J., & Safro, I. (2011). Algebraic distance on graphs. *SIAM Journal on Scientific Computing, 33*(6), 3468–3490.

Cobzaş, Ş, Miculescu, R., & Nicolae, A. (2019). *Lipschitz functions*. Springer.

Daswani, M., Sunehag, P., & Hutter, M. (2013). Q-learning for history-based reinforcement learning. In *Asian Conference on Machine Learning (PMRL)* (pp. 213–228).

Deza, M. M., & Deza, E. (2009). *Encyclopedia of distances*. Springer.

Dolgov, D., & Durfee, E. (2006). The effects oflocality and asymmetry in large-scale multiagent mdps. In *Coordination of large-scale multiagent system* (pp. 3–26).

Dong, M., Yang, X., Wu, Y., Xue, J.H. (2018). Metric learning via maximizing the lipschitz margin ratio. arXiv preprint arXiv:180203464.

Driessens, K., Ramon, J., & Gärtner, T. (2006). Graph kernels and gaussian processes for relational reinforcement learning. *Machine Learning, 64*(1–3), 91–119.

Entringer, R., Douglas, C., Jackson, E., & Synder, D. A. (1976). Distance in graphs. *Czechoslovak Mathematical Journal, 26*(2), 283–296.

Gao, X., Xiao, B., Tao, D., & Li, X. (2010). A survey of graph edit distance. *Pattern Analysis and applications, 13*(1), 113–129.

Goddard, W., & Oellermann, O. R. (2011). Distance in graphs. *Structural Analysis of Complex Networks* (pp. 49–72). Birkhäuser.

Goldberg, A., & T R,. (1993). A heuristic improvement of the bellman-ford algorithm. *Report STAN-CS-93-1464* (pp. 1–5). Dept. of Computer Sc.: Stanford University, Stanford University.

Gottlieb, L. A., Kontorovich, A., & Krauthgamer, R. (2014). Efficient classification for metric data. *IEEE Transactions on Information Theory, 60*(9), 5750–5759.

Goyal, P., & Ferrara, E. (2018). Graph embedding techniques, applications, and performance: A survey. *Knowledge-Based Systems, 151,* 78–94.

Graham, R. L., Hoffman, A. J., & Hosaya, H. (1977). On the distance matrix of a directed graph. *Journal of Graph Theory, 1,* 85–88.

Hakimi, S., & Yau, S. (1965). On the distance matrix of a directed graph. *Quarterly of Applied Mathematics, 22,* 305–317.

He, K., Banerjee, B., & Doshi, P. (2020). *Cooperative-competitive reinforcement learning with history-dependent rewards*. arXiv preprint arXiv:2010.08030.

Hjaltason, G. R., & Samet, H. (2003). Index-driven similarity search in metric spaces (survey article). *ACM Transactions on Database Systems (TODS), 28*(4), 517–580.

Howard, R. (1960). *Dynamic programming and Markov processes*. John Wiley & Sons.

Jia, H., Ym, Cheung, & Liu, J. (2015). A new distance metric for unsupervised learning of categorical data. *IEEE Transactions on Neural Networks and Learning Systems, 27*(5), 1065–1079.

Klein, D., & Randić, M. (1993). Resistance distance. *Journal of Mathematical Chemistry, 12*(1), 81–95.

Kubat, M. (2017). *An introduction to machine learning*. Springer.

Kyng, R., Rao, A., Sachdeva, S., & Spielman, D. A. (2015). Algorithms for Lipschitz learning on graphs. In *Conference on Learning Theory* (pp. 1190–1223).

Majeed, S. J., & Hutter, M. (2018). On q-learning convergence for non-markov decision processes. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence (IJCAI)* (pp. 2546–2552).

Mustăţa, C. (2001). Extensions of semi-lipschitz functions on quasi-metric spaces. *Revue d'analyse numérique et de théorie de l'approximation*, *30*(1), 61–67.

Mustăţa, C. (2002). On the extremal semi-lipschitz functions. *Revue d'analyse numérique et de théorie de l'approximation*, *31*(1), 103–108.

N'Guyen, S., Moulin-Frier, C., & Droulez, J. (2013). Decision making under uncertainty: a quasimetric approach. *PloS one, 8*(12), e83411.

Nickel, M., Murphy, K., Tresp, V., & Gabrilovich, E. (2015). A review of relational machine learning for knowledge graphs. *Proceedings of the IEEE, 104*(1), 11–33.

Ou, M., Cui, P., Pei, J., Zhang, Z., & Zhu, W. (2016). Asymmetric transitivity preserving graph embedding. In: In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (pp. 1105–1114).

Puterman, M. (1994). *Markov decision processes*. John Wiley & Sons.

Rao, A. (2015). *Algorithms for Lipschitz Extensions on Graphs*. Yale University.

Romaguera, S., & Sanchis, M. (2000). Semi-Lipschitz functions and best approximation in quasi-metric spaces. *Journal of Approximation Theory, 103*(2), 292–301.

Shaw, B., Huang, B., & Jebara, T. (2011). Learning a distance metric from a network. In *Advances in Neural Information Processing Systems* (pp. 1899–1907).

Sigaud, O., & Oe, Buffet. (2013). *Markov decision processes in artificial intelligence*. Wiley & Sons.

Singh, S., & Cohn, D. (1998). How to dynamically merge Markov decision processes. *Advances in Neural Information Processing Systems, 10,* 1057–1063.

Sutton, R., & Barto, A. (2017). *Introduction to reinforcement learning*. MIT Press.

von Luxburg, U., & Bousquet, O. (2004). Distance-based classification with Lipschitz functions. *Journal of Machine Learning Research, 5,* 669–695.

Waradpande, V., Kudenko, D., & Khosla, M. (2020). *Graph-based state representation for deep reinforcement learning* (pp. 1–17), arXiv:200413965.

Wu, Y., Jin, R., & Zhang, X. (2016). Efficient and exact local search for random walk based top-k proximity query in large graphs. *IEEE Transactions on Knowledge and Data Engineering, 28*(5), 1160–1174.

Xia, P., Zhang, L., & Li, F. (2015). Learning similarity with cosine similarity ensemble. *Information Sciences, 307,* 39–52.

Xu, Z., Ke, Y., Wang, Y., Cheng, H., & Cheng, J. (2012). In: Proceedings of the 2012 ACM SIGMOD international conference on management of data. A model-based approach to attributed graph clustering (pp. 505–516).

Zhang, T., Gao, Y., Chen, L., Chen, G., & Pu, S. (2019). Efficient similarity search on quasi-metric graphs. *IEEE Access, 7,* 101496–101512.