TRACE: Trajectory Clustering for Explainability

Aimilia Palaska[1] , **Eli Verwimp**[1] , **Pieter Libin**[1,2]
[1] Artificial Intelligence Lab, Vrije Universiteit Brussel, Brussels, Belgium
[2] Data Science Institute, UHasselt, Hasselt, Belgium
{Emily.Palaska, Eli.Verwimp, Pieter.Libin}@vub.be

Abstract

Multi-Objective Reinforcement Learning (MORL) provides a powerful framework for sequential decision-making under competing objectives, but the resulting policies often remain difficult to interpret. Existing explainability approaches typically focus on isolated state-action pairs or rely on black-box representations, offering limited insight into the broader behavioral strategies that produce different trade-offs on the Pareto front. This limitation is particularly problematic in high-stakes domains, where decision makers must understand not only the outcomes of a policy, but also the sequence of decisions that lead to them. To address this challenge, we introduce *TRACE* (Trajectory Clustering for Explainability), a method for interpreting MORL solutions through trajectory-level behavioral analysis. TRACE clusters trajectories according to behavioral similarity by modeling state-conditioned action distributions and comparing them through interpretable distance metrics. Using k-medoids clustering, the method identifies coherent groups of trajectories that represent distinct behaviors, connecting these patterns to regions of the Pareto front and generating intuitive explanations through trajectory visualizations and compact strategy summaries. We evaluate TRACE on two MORL benchmark environments using both ground-truth Pareto-optimal trajectories and trajectories generated by a state-of-the-art MORL algorithm. Our results show that TRACE effectively summarizes thousands of trajectories into a limited number of clusters, providing actionable insights into how different decision-making strategies realize distinct multi-objective trade-offs.

1 Introduction

Real-world decision-making is rarely simple and a single criterion, or objective, almost never suffices. Whether mitigating a crisis, allocating resources, or balancing societal trade-offs, policymakers must simultaneously optimize competing goals [Roijers *et al.*, 2013]. For example, the decision to limit social contact during an epidemic can be influenced by both infection rates and the societal burden of enforcing mitigation measures [Reymond *et al.*, 2024].

The inherent complexity of decision making has driven significant progress in reinforcement learning (RL) as a decision support framework, where agents are trained to identify policies that can assist expert decision makers in complex sequential decision-making problems. Building on this foundation, multi-objective reinforcement learning (MORL) is particularly well suited to settings in which decisions must account for multiple, often conflicting, goals. Rather than prescribing a single course of action, MORL allows decision makers to explore trade-offs between objectives and choose policies that best reflect their priorities. This way, the trained agent can support rather than replace human experts, who bring additional domain knowledge and practical experience, by highlighting high-quality policies in situations where manually navigating the full decision space is infeasible.

A critical gap remains between MORL methods and decision-making applicability. State-of-the-art MORL algorithms predominantly rely on opaque, black-box models, such as neural networks [Röpke *et al.*, 2025; Reymond *et al.*, 2022], raising serious concerns in high-stakes domains like healthcare. More importantly, these approaches typically communicate decisions only at the level of isolated state-action mappings, offering limited insight into the broader behavioral patterns that emerge under a given preference specification. While such local decisions may indicate the optimal action for a particular state, they fail to convey how sequences of decisions unfold over time to realize a desired trade-off and how they relate to alternative similar courses of action. Attempting to manually inspect all preferences and potential paths quickly becomes infeasible, as analysts are faced with hundreds or thousands of state-actions pairs, obscuring meaningful patterns and making trade-offs difficult to interpret.

To address this challenge, we introduce *TRACE* (Trajectory Clustering for Explainability): a method designed to make MORL solutions interpretable and actionable. In RL, a trajectory is the sequence of visited states and executed actions for a particular execution of a policy. TRACE clusters trajectories obtained from the Pareto set of optimal policies, based on behavioral similarity rather than outcome alone. By leveraging both trained policies and ground truth algorithms to generate trajectories that lead to Pareto front points, we analyze and group these trajectories to uncover underlying behavioral pat-

terns. Our approach emphasizes white-box representations of trajectories, enabling straight-forward human interpretation.

By shifting the focus from vectors of reward values to trajectories, TRACE reveals when and where certain decisions drive different outcomes, allowing us to identify which decisions contribute to specific regions of the Pareto front. Through these insights, TRACE generates intuitive visualizations and explanations, including temporal graphs and concise decision trees. These outputs could equip decision makers with clear, structured guidance, making it easier to navigate trade-offs and select strategies aligned with their objectives.

We evaluate TRACE on two MORL benchmark environments, namely Deep Sea Treasure by Vamplew *et al.* [2011] and MineCart by Abels *et al.* [2019]. We introduce a new explainability-oriented benchmark derived from the Minecart environment, named MineTrain, through targeted modifications designed to ease the computation of Pareto optimal trajectories without sacrificing the environment's inherent complex dynamics. Furthermore, we contribute deterministic algorithms for computing the ground-truth optimal trajectories in these environments, enabling more rigorous evaluation and benchmarking of explainability methods against known optimal behaviors.

Through this evaluation, we demonstrate TRACE's ability to extract trajectories and group them into meaningful patterns for both simple and complex settings. Across experiments, TRACE effectively clusters large numbers of trajectories into groups that reveal meaningful behavioral patterns. In both environments, we introduce ground-truth baseline algorithms to generate all possible pareto-optimal trajectories, alongside trajectories generated by policies trained with the IPRO algorithm created by Röpke *et al.* [2025]. We show that TRACE successfully manages to encapsulate behavior in both cases, further underlining its flexibility. To support reproducibility and further research, we also release our implementation as open-source code[1].

2 Background

2.1 Multi-Objective Markov Decision Processes

Sutton and Barto [2018] provide a mathematical framework for modeling sequential decision-making problems with more than one conflicting goal under uncertainty, called Multi-Objective Markov Decision Processes (MOMDPs). A MOMDP is defined by the tuple:

$$\mathcal{M} = (\mathcal{S}, \mathcal{A}, T, R, \gamma), \tag{1}$$

where $\mathcal{S}$ denotes the state space, $\mathcal{A}$ the action space, $T(\mathbf{s}' \mid \mathbf{s}, a)$ the transition probability function describing the dynamics of the environment, $R(\mathbf{s}, a)$ the reward function describing $d > 1$ objectives, and $\gamma \in (0, 1]$ the discount factor.

At each discrete time step t, the agent observes a state $\mathbf{s} \in \mathcal{S}$, selects an action $a \in \mathcal{A}$ according to the agent's policy $\pi(a|s)$, and receives a reward $\mathbf{r} = R(\mathbf{s}, a)$, after which the system transitions to a new state $\mathbf{s}'$ according to the transition dynamics T. Performing multiple steps and collecting the sequences of states, actions and respective rewards constitutes a trajectory τ of length T:

$$\tau = \{(s_1, a_1, r_1) \dots (s_T, a_T, r_T)) \mid \pi(a|s)\} \tag{2}$$

For a given trajectory τ with a reward sequence $\{\mathbf{r}_t\}_{t=0}^{T}$, where T denotes the horizon, the corresponding discounted return, also referred to as the value vector, is defined as

$$\mathbf{v}_\tau = \sum_{t=0}^{T} \gamma^t \, \mathbf{r}_t \in \mathbb{R}^d. \tag{3}$$

This vector represents the cumulative discounted performance of the trajectory across all objectives simultaneously.

2.2 Pareto Optimality

Compared to approaches that combine objectives into a single scalar reward using weighted sums, the Pareto front enables decision makers to explicitly select their preferred trade-off between objectives without making prior assumptions about the relative importance or scale of the different components in reward vector. Formally, a value vector $\mathbf{v} \in \mathbb{R}^d$ is said to strictly dominate another vector $\mathbf{u} \in \mathbb{R}^d$ if

$$\begin{cases} v_i \succeq u_i & \forall i \in \{1, \dots, d\}, \\ v_j \succ u_j & \text{for at least one } j. \end{cases} \tag{4}$$

A value vector is Pareto-optimal if it is not strictly dominated by any other feasible vector. The set of all such non-dominated vectors forms the Pareto front.

Finally, it is important to distinguish between non-dominated value vectors and Pareto-optimal policies. A Pareto-optimal value vector corresponds to a performance outcome in the objective space, whereas Pareto-optimal policies refer to the set of policies that attain these outcomes. In general, distinct policies or even distinct trajectories may yield the same value vector, especially in stochastic environments. As a result, the set of Pareto-optimal policies is typically larger than the Pareto front itself.

3 Related Work

Explainable reinforcement learning (XRL) aims to improve transparency by revealing the reasoning behind an agent's decisions and supporting human understanding of learned policies. Despite growing interest in explainability, its application to explainable multi-objective reinforcement learning (XMORL) remains limited.

A common line of work in XRL explains policies by identifying states that have a strong influence on reward outcomes. Luss *et al.* [2023] identify important states by exploiting locality relationships, defined via likelihood-based distances between states, and demonstrate that state-based explanations are more informative to users than action-level aggregations. Li *et al.* [2025] propose a model-agnostic framework based on Shapley values, quantifying how individual state features contribute to an agent's selected action in single-objective RL. By clustering the resulting high-dimensional Shapley vectors using k-means, they identify distinct regions in the

[1] https://github.com/ANTICIPATE-FWO/trace

state space associated with different decision strategies and derive inter-cluster decision boundaries.

In multi-objective settings, the Pareto-front Analyzer (PAN) is introduced in [Ulrich, 2013], which clusters Pareto-optimal solutions using a bi-objective evolutionary algorithm, considering both the decision and the objective space. Since it is not designed for MOMDPs, the decision space is an algorithmic-agnostic distance function between contents of the solution. Building on this idea, Osika *et al.* [2024] adapt PAN to MORL by representing policy behavior using the most important states identified by HIGHLIGHTS and comparing them via a Frobenius-like distance metric. Complementarily, Hu *et al.* [2024] consider policy distance in the context of multi-agent reinforcement learning, by representing policies as action distributions conditioned on observations through an autoencoder and addressing state-dependence through distance metrics.

Finally, a recent and promising line of work in XMORL by [Rajapakse *et al.*, 2026] explores trajectory clustering for summarizing and explaining policy behavior. In particular, they generate trajectories using a preference-conditioned Multi-Objective Decision Transformer MODT(P) [Zhu *et al.*, 2023], which are represented as sequences of visited states. These are embedded using the attention-based transformer of the trained policy and clustered according to their returns. This demonstrates that trajectory clustering is a viable and insightful direction for MORL explainability. At the same time, this approach primarily relies on state-level representations and does not explicitly incorporate actions as part of the behavioral signature. Moreover, the use of black-box embedding models for feature extraction, while effective for clustering, introduces an additional layer of abstraction that is not always aligned with the goal of transparent interpretability. In contrast, our method complements this direction by directly integrating actions into the analysis and by emphasizing more interpretable trajectory representations, while still benefiting from the insights that clustering-based explanations provide.

Relevant work in the field of policy interpretability has been done using interpolation techniques. PGMORL [Xu *et al.*, 2020] applies interpolation in the parameter space, in the context of prediction-guided optimization. It includes a process called Pareto analysis, which clusters the parameters of fitted monotonic hyperbolic models using k-means into families of similar objective-preference with the goal of interpolating them to achieve a continuous Pareto front. They hint at the fact that each of these families is a disjoint pattern, however their goal is to leverage the policy improvement models for Pareto front refinement, instead of explainability.

In the direction of explainability, several methods aim to extract interpretable structures from complex models and clustering outcomes. Tan et el. [Tan *et al.*, 2026] propose a framework that distills neural networks through symbolic regression, transforming pairs of inputs and outputs into explicit mathematical expressions. Focusing more directly on explaining clustering outcomes, [Tutay and Somech, 2023] introduce *Cluster Explorer*, a technique designed to explain black-box clustering results using generalized frequent itemset mining by [Srikant and Agrawal, 1997]. The method partitions each feature dimension into value ranges and identifies frequent patterns that characterize clusters, but leaves scaling in high-dimensional settings an open question. Similarly, [Ellis *et al.*, 2021] propose a feature-importance measure tailored to clustering. Here, features that cause larger cluster shifts when permuted are considered more influential in defining the clustering structure.

To the best of our knowledge, no prior work has explored clustering in MORL using white-box trajectory representations that explicitly capture behavioral aspects considering both state and action sequences, particularly for interpreting different trade-off options among Pareto-optimal trajectories.

4 TRACE

A MORL policy is defined both in terms of its expected return and its behavior, namely how it acts in different states. For this reason, we consider the behavior and return space to be related but distinct. Behavior describes an agent's actions based on the states they encounter, while the objective space only looks at how much an agent scored in each dimension of the return. Our goal is twofold: (a) summarize the high-dimensional behavior space into clear patterns that tie closely to human-level interpretation and (b) associate these behavioral patterns to regions of the Pareto front. We split our algorithm into four parts, visualized in Figure 1 in the form of a high level pipeline. Firstly, we collect the trajectories that will be analyzed. We then model the agent's behavior by reshaping the trajectories into similarity features for comparison. We apply the k-medoids clustering algorithm on the pre-computed behavioral features to group trajectories that tend to select the same actions in the same states. Finally, we propose methods for explaining each cluster by extracting their shared behavioral patterns in a human-interpretable manner.

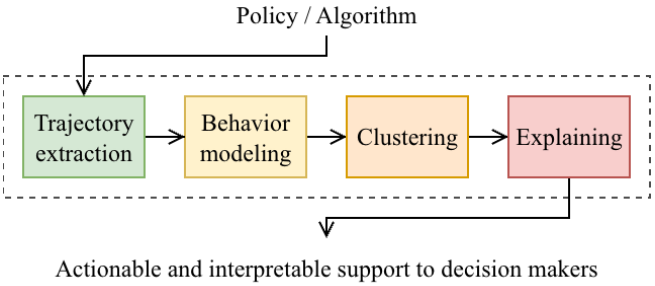


Figure 1: The key parts of the TRACE method in a high-level pipeline overview.

4.1 Trajectory collection

TRACE operates on a set trajectories as its primary data input. Each trajectory concerns a full interaction roll-out between the agent and the environment, meaning the sequence of observed states, selected actions, and corresponding rewards. Within this framework, we assume that the provided trajectories lead to a point in the Pareto front, or to approximation of it known as the coverage set. A key advantage of this setup is that it does not impose assumptions on how the Pareto front was obtained. As long as trajectories can be loaded in

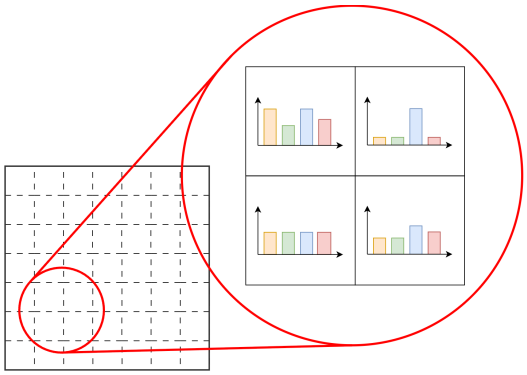


Figure 2: Intuitive visualization of the state-conditioned empirical action distribution $\hat{P}_{\tau_i}$. Each state in $\mathcal{S}$, represented here in the form of a grid, consists of a distinct distribution derived from its observed actions.

a structured format, through a trained policy, a ground truth deterministic algorithm or even synthesized data-points, the analysis can be applied.

4.2 Behavior Modeling

Addressing the question of when two trajectories should be considered behaviorally similar is non-trivial, as it requires summarizing large amounts of trajectory data, often concerning high-dimensional or continuous state spaces, into a compact representation that can be compared across policies.

To incorporate state-dependent structure into the behavioral representation, we condition actions on the states in which they are performed, as seen in Figure 2. For a given trajectory τ, we define a state-conditional empirical action distribution,

$$\hat{P}_\tau(a \mid \mathbf{s}) = \frac{N_\tau(\mathbf{s}, a) + \alpha}{\sum_{a' \in \mathcal{A}} \left(N_\tau(\mathbf{s}, a') + \alpha\right)} \tag{5}$$

where $N_\tau(\mathbf{s}, a)$ denotes the number of times action a was taken in state $\mathbf{s}$ along trajectory τ, and $\alpha > 0$ is a smoothing parameter. This distribution can be interpreted as a trajectory-specific behavioral policy, analogous to the policy $\pi(a \mid \mathbf{s})$ commonly used in reinforcement learning [Sutton and Barto, 2018]. We set $\alpha = 0.5$, corresponding to a symmetric Dirichlet prior over the action probabilities [Bishop and Nasrabadi, 2006]. This smoothing is useful for sparse trajectories, where many state-action pairs may be unobserved or rarely visited.

4.3 Clustering

To perform the clustering of trajectories, we employ the k-medoids partitioning algorithm [Kaufman, 1990]. Given the set of ground-truth trajectories, the algorithm partitions them into k distinct clusters by selecting representative trajectories, referred to as medoids, directly from $\mathcal{T}$. In every iteration of the algorithm, each trajectory is assigned to the cluster containing the closest medoid, according to a predefined trajectory similarity metric. The medoids are iteratively updated by replacing candidate representatives as long as doing so reduces the overall intra-cluster dissimilarity. This optimization process continues until convergence, yielding clusters whose members are maximally similar to their corresponding representative trajectory.

Unlike centroid-based clustering approaches, k-medoids operates by selecting representative samples directly from the dataset rather than relying on the arithmetic mean of cluster elements. As a result, it can better accommodate arbitrary and custom-defined dissimilarity metrics, including non-Euclidean measures and metrics that are not necessarily consistent with mean-based representations, as discussed in [Kaufman, 1990]. Furthermore, because cluster centers are restricted to actual observed samples, k-medoids is substantially more robust to noisy data and outliers, which might otherwise distort centroid estimates. This property is particularly well-suited for trajectory clustering, where behavioral similarity may not be adequately captured by simple geometric averages.

To derive the distance features from empirical distributions, a direct approach consists of computing the Frobenius distance between the respective action distributions. Due to trajectory sparsity, meaning that the number of visited states tends to be a small fraction of all the possible ones, we only consider overlapping states, i.e. states visited by both trajectories. States visited by only one trajectory do not provide sufficient evidence to reliably compare the corresponding action-selection behavior.

Formally, let τ_i, τ_j be two trajectories and $\hat{P}_{\tau_i}(a \mid \mathbf{s})$ and $\hat{P}_{\tau_j}(a \mid \mathbf{s})$ denote their respective action distributions for a given state $\mathbf{s} \in \mathcal{S}$. Aggregating over all overlapping states $\mathcal{S}_{ij}$, the global Frobenius distance is given by

$$d_F(\tau_i, \tau_j) = \frac{1}{|\mathcal{S}_{ij}|} \sqrt{\sum_{\mathbf{s} \in \mathcal{S}_{ij}} \sum_{a \in \mathcal{A}} \left(\hat{P}_{\tau_i}(a \mid \mathbf{s}) - \hat{P}_{\tau_j}(a \mid \mathbf{s})\right)^2} \tag{6}$$

This metric aligns together trajectories that exhibit similar action distributions in the same regions of the state space, effectively grouping together policies with analogous behavioral tendencies. Finally, normalization is applied in terms of the amount of common visited states to make this metric meaningful and comparable across the whole set of trajectories.

The Frobenius norm compares all possible actions in all states, including those with a low probability. Many small differences for low probability actions can lead to a high norm, even when policies always take the same action. To avoid this issue, we propose a direct trajectory overlap metric. The intuition is that similarly behaving strategies should tend to take the same action in the same state. To quantify this, we measure how frequently two policies agree, only considering the most likely action in shared states. Formally, let:

$$a_{i,\mathbf{s}} = \arg\max_{a \in \mathcal{A}} \hat{P}_i(a \mid \mathbf{s}) \tag{7}$$

be the action with the highest probability in a given state $\mathbf{s} \in \mathcal{S}$ for a trajectory τ_i. The agreement rate between policies τ_i and τ_j is defined as

$$\text{Agree}(\tau_i, \tau_j) = \frac{1}{|\mathcal{S}_{ij}|} \sum_{s \in \mathcal{S}_{ij}} \mathbf{1}\left[a_{i,\mathbf{s}} = a_{j,\mathbf{s}}\right], \tag{8}$$

where $\mathbf{1}[\cdot]$ denotes the indicator function. Finally, to allow minimization procedures for both metrics, we convert to a dissimilarity:

$$d_A(\tau_i, \tau_j) = 1 - \text{Agree}(\tau_i, \tau_j). \tag{9}$$

The agreement metric directly captures behavioral consistency by emphasizing whether two policies prescribe the same decision in the same situation, thereby providing a more straightforward measure of similarity for clustering.

4.4 Strategy explanation

After determining the number of behavioral clusters and performing the partitioning, it is vital to find ways to communicate the patterns in a clear and human-interpretable way. In other words, we have to focus on the differences between clusters and extract conclusions from the whole analysis.

One approach would be to visualize all trajectories associated with a cluster, across the entire state space in the form of a map and let users identify distinguishable features of them. This method gives a clear representation of what each strategy would look like based on what action tends to be picked at each region of the state space. Evidently, it is more compatible to environments that employ a sort of spatial representation, such as a map or a grid. For a more temporally grounded perspective, an alignment graph can visualize the action selected at each timestep with an action-coded color map. Such temporal structure may reveal distinctions that are not immediately apparent in a spatial visualization alone, particularly when the significance of an action depends not only on the state itself but also on the timing at which it occurs. Lastly, we also examine qualitative cluster-level statistics that can be tailored to the environment under study. This provides a flexible layer of explanation, as interpretability depends inherently on context.

Across all of these components, the goal is to form a clear and coherent picture of each cluster of trajectories by capturing both its spatial and temporal characteristics, summarizing the range of rewards it may produce, and describing its decision structure through simple and interpretable rules. In this way, we aim to provide a compact yet informative explanation of thousands of data points, allowing decision makers to progressively narrow down their options in a structured and understandable manner.

5 Environments

5.1 Deep Sea Treasure

The Deep Sea Treasure [Vamplew *et al.*, 2011] environment is a grid-based navigation task in which an agent controls a submarine exploring an underwater map. The state space $\mathbf{s} = (x, y) \in \mathcal{S}$ where $x, y \in \{0, 1, \ldots, 11\}$ consists of the discrete grid coordinates of the submarine, representing its position in the environment. The action space is given by $\mathcal{A} = \{\text{up}, \text{down}, \text{left}, \text{right}\}$, corresponding to deterministic movements in the grid, subject to boundary and obstacle constraints.

This environment encompasses a multi-dimensional reward function, reflecting the multi-objective nature of the problem. At each time step, the agent receives $\mathbf{r} = \big(r_1,\ r_2\big)$ where r_1 is a sparse reward obtained only when a treasure is reached (with its value depending on the treasure's depth), and $r_2 = -1$ penalizes each time step to encourage shorter paths. The trade-off between collecting high-value treasures, which are typically located deeper, and minimizing time induces a Pareto front over policies.

To collect all the trajectories that lead to a reward with minimal steps, we use a constrained breadth-first traversal that enumerates all monotonic shortest-path trajectories from the start state $\mathbf{s}_0 = (0, 0)$ to every treasure state. In this context, monotonic refers to only moving using the `down` and `right` movements, since any other path would be sub-optimal time-wise. The detailed algorithm can be found in Appendix A. To the best of our knowledge, this is the first deterministic formalization of ground truth trajectory collection for this environment.

5.2 MineTrain

MineCart is an established benchmark MORL environment that models a continuous resource collection and management problem in which an agent controls a cart that navigates between resource mines and a base. The state space $\mathcal{S}$ consists of the variables:

$$\mathbf{s} = \big(x, y, v, \sin(\theta), \cos(\theta), \ell_1, \ell_2\big), \tag{10}$$

where $x, y \in [0, 1]$ denote the cart position, $v \in [0, 1]$ its absolute velocity, $\sin(\theta), \cos(\theta) \in [-1, 1]$ its orientation, and $\ell_1, \ell_2 \in [0, 1.5]$ the quantity of each ore currently stored, constrained by a maximum cart capacity of 1.5. The action space is the discrete set $\mathcal{A} = \{\text{mine, left, right, accelerate, brake, none}\}$, which allows navigation and mining interactions. The reward function is $\mathbf{r} = \big(r_1, r_2, r_{\text{fuel}}\big)$, where $r_1, r_2 \in [0, 1.5]$ correspond to the amount of each ore delivered to the base and $r_{\text{fuel}} \in (-\infty, 0]$ represents the negative fuel cost. Ore quantities at mine locations are sampled from mine-specific and ore-specific normal distributions $N(\mu_{m,\ell}, \sigma_{m,\ell})$. Compared to Deep Sea Treasure, this environment exhibits substantially more complex dynamics, since the agent must simultaneously optimize fuel consumption and the extraction of two distinct ore types. Different combinations of ore returns therefore correspond to distinct Pareto-optimal behaviors, even when the total amount of collected ore remains identical. Furthermore, the continuous observation space and stochasticity in ore yields produce a considerably richer and more elaborate Pareto front, making the construction of ground-truth trajectory sets infeasible through exhaustive enumeration and instead requiring heuristic approximations based on cumulative returns rather than full state-action-reward trajectories.

In the absence of reliable ground truth, meaningful evaluation of TRACE becomes infeasible, as explanations cannot be validated against true optimal behavioral patterns, making it impossible to assess their semantic consistency. To address this limitation, we introduce a constrained variant of the environment that preserves the core decision-making challenges while rendering ground-truth computation tractable, and we conduct all experiments on this benchmark as it is better suited for studying explainability.

The modified environment, called *MineTrain*, introduces linear trails connecting the base and mine locations. The agent may accelerate, brake, and mine while traversing these trails, whereas cart rotation is permitted only on the base and mine locations. Stochastic mining dynamics are removed by replacing ore distributions with yields equal to the mean of the original distributions, making the set of optimal trajectories deterministic and tractable. To compute the ground-truth Pareto front, we first enumerate all feasible mine visitation paths of every possible length starting and ending at the base. For each path, mines are ordered to minimize total traveled distance and therefore fuel consumption. Since all mine locations belong to the convex hull in the configuration we consider, optimal routes follow the same ordering as the convex hull traversal [MIT Urban Operations Research, nd]; both forward and reverse traversals are included for completeness.

Fuel-efficient acceleration strategies are then derived analytically. As in the original environment, each timestep corresponds to a fuel penalty of $\texttt{FUEL_IDLE} = 0.005$, while acceleration gives an additional $\texttt{FUEL_ACC} = 0.025$ and increases speed by $\texttt{SPEED_ACC} = 0.0075$. For a trail of length l, there exists an optimal number of acceleration steps $k_{opt} \in \mathbb{N}$ minimizing fuel consumption. We assume that starting speed is zero, and try all the scenarios for $k = \{1, 2, \dots k_{max}\}$, where k_{max} expresses the maximum number of acceleration steps and is calculated analytically as:

$$k_{max} = \left\lceil \frac{-1 + \sqrt{1 + \frac{8l}{\texttt{SPEED_ACC}}}}{2} \right\rceil . \tag{11}$$

After deriving the acceleration strategy, we also consider the times the cart will perform the mining action in each location. It is important to note that there is maximum cart capacity of $c = 1.5$ as well as a mining fuel penalty of $\texttt{FUEL_MINE} = 0.05$. Therefore, a trajectory's value vector with more mining actions will not automatically dominate less mining, even on the same path. For this reason, we consider all the combinations mining steps, such that the cart's capacity is not surpassed but also guarantee that if a mine is visited, it will be mined at least once. Doing so avoids redundant traveling to mines without trying to obtain ores, which would be sub-optimal fuel-wise.

6 Preliminary Results

6.1 Quantitative metrics

To enable a quantitative comparison between different trajectory partitioning methods, we introduce a metric that evaluates the quality of clusters. Intuitively, an effective clustering yields coherent groups, each describing a distinct behavioral pattern that can be meaningfully interpreted. To do so, we assume that similar behavior is exhibited by taking decisive actions in similar states. The decisiveness of a distribution can be measured using the average deviation V from the uniform distribution (where every action has probability $\frac{1}{|\mathcal{A}|}$):

$$V(\hat{P}_\tau(A \mid S)) = \frac{1}{|\mathcal{A}|} \sum_{\alpha \in \mathcal{A}} (\hat{P}_\tau(\alpha \mid \mathbf{s}) - \frac{1}{|\mathcal{A}|})^2. \tag{12}$$

Higher values indicate a sharper distribution, with $V = 0$ if the action distribution is uniform. A second option is using the distributions entropy:

$$H(\hat{P}(A|S)) = -\sum_{\alpha \in A} \hat{P}(\alpha|S) \log(\hat{P}(\alpha|S) \tag{13}$$

which measures the amount of information in a random draw from the distribution at hand. Entropy equals 1 for uniform distribution and 0 for entirely decisive distributions (because no information is learned by drawing from this distribution). Next, we define the set $\mathcal{V}$ as the set of all states visited by a trajectory part of a set of trajectories $\mathcal{T}$:

$$\mathcal{V}_c = \{s \in S \mid \exists \tau \in \mathcal{T}_c : s \in \tau\} \tag{14}$$

Using the two decisiveness measures and the set of visited states, we can define the decisiveness of cluster c, with trajectories $\mathcal{T}_c$ as:

$$D_c = \frac{1}{|\mathcal{V}_c||T_c|} \sum_{s \in \mathcal{V}_\rfloor} \sum_{\tau \in \mathcal{T}_\rfloor} V(\hat{P}_\tau(A \mid S)) \tag{15}$$

when using the average deviation and

$$D_c^{(e)} = \frac{1}{|\mathcal{V}_c||T_c|} \sum_{s \in \mathcal{V}_c} \sum_{\tau \in \mathcal{T}_c} \Big(1 - H(\hat{P}_\tau(A \mid S))\Big) \tag{16}$$

for the entropy based metric. Both metrics will be high if for many of the states in $\mathcal{V}$, all trajectories are decisive. When many action distributions in the states of $\mathcal{V}$ are close to uniform, both metrics will have values near zero.

6.2 Experiments

For the preliminary experiments, we ran the ground truth algorithms on both environments and documented 3601 trajectories for DST and 1120 trajectories for MineTrain, all yielding value vectors that belong to the Pareto front of the respective settings. Additionally, we trained the IPRO algorithm, replicating the hyperparameters used in the publication [Röpke *et al.*, 2025], and gathered an additional 446 coverage set trajectories for MineTrain. IPRO experiments for trajectory collection in Deep Sea Treasure will be presented in future work.

In Table 1 we present the initial results on the proposed metrics, including a random label clustering as an ablation reference point. Results suggest the clustering is indeed increasing decisiveness but improvement is in some cases minimal, which shows the potential to experiment with different clustering algorithms and metrics. Inevitably, random clustering does introduce some decisiveness in the sense that some states will end up not getting visited, but still remains below the proposed method.

For a more visual grasp at the results, we present the temporal alignment plots of the MineTrain trajectories in Figure 3, separated into 5 groups using the agreement metric on the k-medoids algorithm. The visualization shows how this representation of clusters offers intuitive insights. We can see that the clustering aligned the chosen actions more clearly in

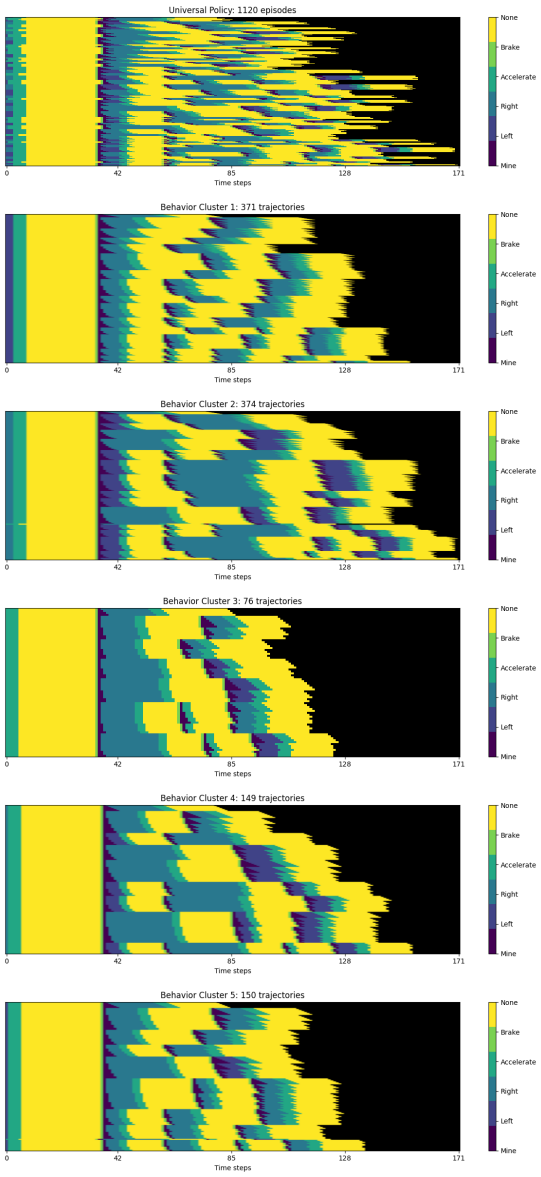


Figure 3: Temporal alignment plot, all the trajectories from MineTrain's ground truth policy (top) in comparison with final clusters, using k-medoids and the agreement features.

the clusters rather than the full set of trajectories, and specifically split them based on the direction of rotation, while also being influenced by the trajectory horizon. Similarly, state space visualization of decisiveness in Figure 4 explains areas where the trajectories converge in their chosen action.

7 Future research

We present this method as a work in progress. While our preliminary results indicate a promising direction for improving the explainability of MORL policies, several aspects still require further refinement. Future work involves introducing better suited behavior features and calibrating the evaluation metrics. Additionally, the dimensionality challenge in environments with large, continuous or infinite state spaces will be addressed by incorporating state importance into the computation of clustering features. Finally, direct feedback through user studies will be integrated into the evaluation pipeline to more rigorously assess whether the proposed method improves interpretability.

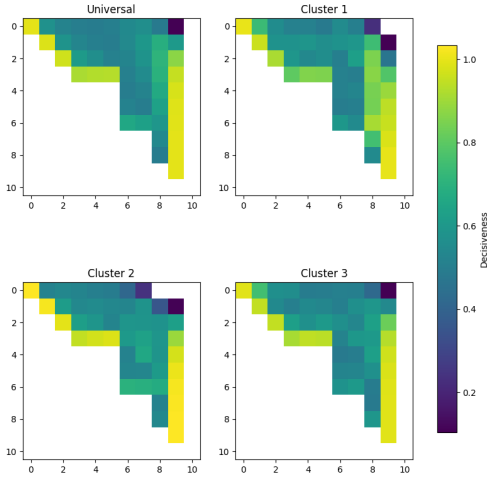


Figure 4: Heatmap of decisiveness per state for DST, states that were not visited are shown in white. This structure is more difficult to visualize in MineTrain due to the high-dimensional state space and will be included in future work.

Env. (Policy)	Method (metric)	Avg. D_c	Avg. $D_c^{(e)}$
DST (GT)	TRACE (f)	**0.34**	**0.68**
	No clustering	0.32	0.66
	Random	0.32	0.65
MT (GT)	TRACE (a)	**0.89**	**1.15**
	No clustering	0.16	0.22
	Random	0.4	0.5
MT (IRPO)	TRACE (a)	**0.59**	**0.73**
	No clustering	0.13	0.17
	Random	0.44	0.61

Table 1: Quantitative metrics on ground truth preliminary experiments.

8 Conclusion

In this work, we contributed to the study of explainability in MORL by adapting existing benchmark environments to better support explainability analysis and evaluation. Furthermore, we formulated mathematically grounded ground-truth trajectory collection algorithms for these benchmark environments, enabling more reliable and interpretable assessment of explanation methods. We presented TRACE, a novel methodology, that clusters trajectories using behavioral features into groups of distinct patterns with the goal of supporting decision makers to interpret MORL policies. Through a set of preliminary experiments, we identified behavioral clustering as a promising direction for capturing and analyzing behavioral pattens within MORL settings.

9 Acknowledgments

This research acknowledges funding from the Flemish Government through the AI Research Program. The work was carried out as part of the inter-university project "Artificial Intelligence and Epidemic Modeling to Prepare Hospitals for the next Respiratory Pathogen with Pandemic Potential (ANTICIPATE)", a collaboration between VUB, UAntwerp, and UZ Brussel. The authors acknowledge the members of the ANTICIPATE consortium for their constructive discussions and valuable insights that contributed to this research. This project is funded by the Research Foundation – Flanders (FWO) through its Strategic Basic Research (SBO) programme (S002626N). P.L. gratefully acknowledges support from the Research council of the Vrije Universiteit Brussel (OZR-VUB, via grant number OZR3863BOF), and from FWO through the fundamental research project programme (grant number G0A0S26N).

References

[Abels *et al.*, 2019] Axel Abels, Diederik Roijers, Tom Lenaerts, Ann Nowé, and Denis Steckelmacher. Dynamic weights in multi-objective deep reinforcement learning. In *International conference on machine learning*, pages 11–20. PMLR, 2019.

[Bishop and Nasrabadi, 2006] Christopher M Bishop and Nasser M Nasrabadi. *Pattern recognition and machine learning*, volume 4. Springer, 2006.

[Ellis *et al.*, 2021] Charles A Ellis, Mohammad SE Sendi, Eloy Geenjaar, Sergey M Plis, Robyn L Miller, and Vince D Calhoun. Algorithm-agnostic explainability for unsupervised clustering. *arXiv preprint arXiv:2105.08053*, 2021.

[Hu *et al.*, 2024] Tianyi Hu, Zhiqiang Pu, Xiaolin Ai, Tenghai Qiu, and Jianqiang Yi. Measuring policy distance for multi-agent reinforcement learning. In *Proceedings of the 23rd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2024)*, pages 1–9, Auckland, New Zealand, May 2024. International Foundation for Autonomous Agents and Multiagent Systems (IFAAMAS).

[Kaufman, 1990] Leonard Kaufman. Partitioning around medoids (program pam). *Wiley series in probability and statistics*, 344:68–125, 1990.

[Li *et al.*, 2025] Peilang Li, Umer Siddique, and Yongcan Cao. From explainability to interpretability: Interpretable reinforcement learning via model explanations. *Reinforcement Learning Journal*, 6:2256–2270, August 2025.

[Luss *et al.*, 2023] Ronny Luss, Amit Dhurandhar, and Miao Liu. Local explanations for reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 9002–9010, May 2023.

[MIT Urban Operations Research, nd] MIT Urban Operations Research. 6.4.7 The Traveling Salesman Problem. https://web.mit.edu/urban_or_book/www/book/chapter6/6.4.7.html, n.d. Accessed: 2026-05-15.

[Osika *et al.*, 2024] Zuzanna Osika, Jazmin Zatarain Salazar, Frans A. Oliehoek, and Pradeep K. Murukannaiah. Navigating trade-offs: Policy summarization for multi-objective reinforcement learning. In *27th European Conference on Artificial Intelligence*, pages 2919–2926, October 2024.

[Rajapakse *et al.*, 2026] Dilina Rajapakse, Juan C Rosero, and Ivana Dusparic. Trex: Trajectory explanations for multi-objective reinforcement learning. *arXiv preprint arXiv:2603.21988*, 2026.

[Reymond *et al.*, 2022] Mathieu Reymond, Eugenio Bargiacchi, and Ann Nowé. Pareto conditioned networks. In *Proceedings of the 21st International Conference on Autonomous Agents and Multiagent Systems*, pages 1110–1118. IFAAMAS, 2022.

[Reymond *et al.*, 2024] Mathieu Reymond, Conor F Hayes, Lander Willem, Roxana Rădulescu, Steven Abrams, Diederik M Roijers, Enda Howley, Patrick Mannion, Niel Hens, Ann Nowé, et al. Exploring the pareto front of multi-objective covid-19 mitigation policies using reinforcement learning. *Expert Systems with Applications*, 249:123686, 2024.

[Roijers *et al.*, 2013] Diederik M Roijers, Peter Vamplew, Shimon Whiteson, and Richard Dazeley. A survey of multi-objective sequential decision-making. *Journal of Artificial Intelligence Research*, 48:67–113, 2013.

[Röpke *et al.*, 2025] Willem Röpke, Mathieu Reymond, Patrick Mannion, Diederik M Roijers, Ann Nowé, and Roxana Rădulescu. Divide and conquer: Provably unveiling the pareto front with multi-objective reinforcement learning. In *Proceedings of the 24th International Conference on Autonomous Agents and Multiagent Systems*, AAMAS '25, page 1774–1783, Richland, SC, 2025. International Foundation for Autonomous Agents and Multiagent Systems.

[Srikant and Agrawal, 1997] Ramakrishnan Srikant and Rakesh Agrawal. Mining generalized association rules. *Future generation computer systems*, 13(2-3):161–180, 1997.

[Sutton and Barto, 2018] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 2 edition, 2018.

[Tan *et al.*, 2026] Elizabeth SZ Tan, Adil Soubki, and Miles Cranmer. Symtorch: A framework for symbolic distillation of deep neural networks. *arXiv preprint arXiv:2602.21307*, 2026.

[Tutay and Somech, 2023] Sariel Tutay and Amit Somech. Cluster-explorer: an interactive framework for explaining black-box clustering results. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, pages 5106–5110, 2023.

[Ulrich, 2013] Tamara Ulrich. Pareto-set analysis: Biobjective clustering in decision and objective spaces. *Journal of Multi-Criteria Decision Analysis*, 20(5-6):217–234, 2013.

[Vamplew *et al.*, 2011] Peter Vamplew, Richard Dazeley, Adam Berry, Rustam Issabekov, and Evan Dekker. Empirical evaluation methods for multiobjective reinforcement learning algorithms. *Machine learning*, 84(1):51–80, 2011.

[Xu *et al.*, 2020] Jie Xu, Yunsheng Tian, Pingchuan Ma, Daniela Rus, Shinjiro Sueda, and Wojciech Matusik. Prediction-guided multi-objective reinforcement learning for continuous robot control. In *International conference on machine learning*, pages 10607–10616. PMLR, July 2020.

[Zhu *et al.*, 2023] Baiting Zhu, Meihua Dang, and Aditya Grover. Scaling pareto-efficient decision making via offline multi-objective rl. *arXiv preprint arXiv:2305.00567*, 2023.

A Extra material from environments

A frame of each environment is presented in Figure 5. In MineTrain, we plot all possible trails the cart can follow, allowing a small tolerance in the position to account for the continuous state space. Additionally, the pseudo-code for Deep Sea Treasure's ground truth policy is presented in Algorithm 1. Lastly, in Figure 6 we show temporal alignment for the deep sea treasure clustering with the Frobenius distance. Results suggest that the clustered trajectories align more in the first steps, which is intuitively logical due to them visiting the same states in the start and the diverging.

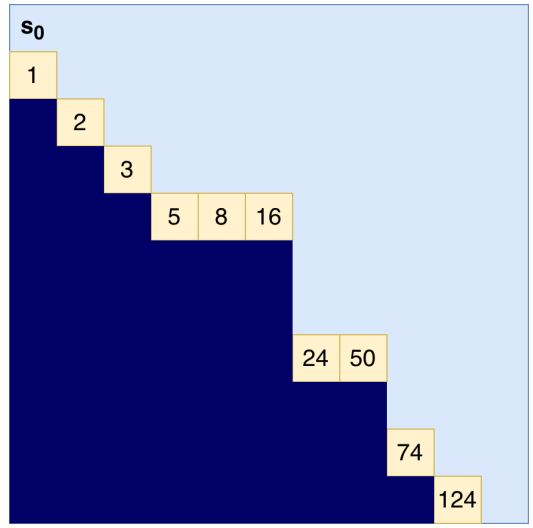
Algorithm 1 DST Ground Truth

Input: MOMDP $\mathcal{M} = (\mathcal{S}, \mathcal{A}, T, R)$, start state $\mathbf{s}_0 \in \mathcal{S}$
Output: Ground truth optimal trajectories set $\mathcal{G}$

Compute the least steps $d(\mathbf{s}_0)$ from $\mathbf{s}_0$ to all states $\mathbf{s} \in \mathcal{S}$.
Identify the set of treasure states:

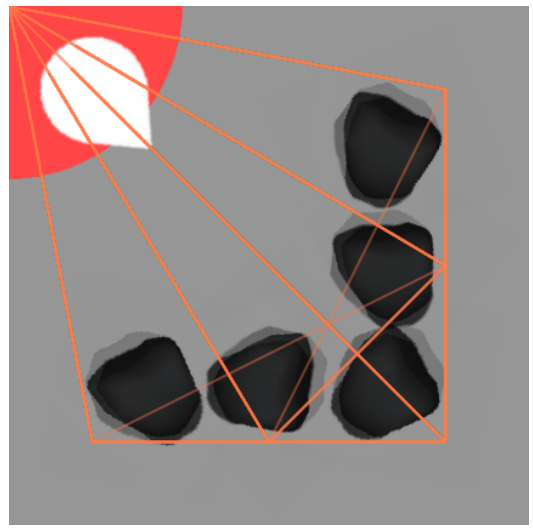
$$\mathcal{S}_{tr} = \{\mathbf{s} \in \mathcal{S} \mid (r_1, r_2) = R(\mathbf{s}, \cdot),\ r_1 > 0\}$$

Initialize $\mathcal{G} \leftarrow \emptyset$
Initialize candidate trajectory set $\mathcal{G}_c \leftarrow \{(\mathbf{s}_0)\}$
while $\mathcal{G}_c \neq \emptyset$ **do**
  Remove a trajectory $\tau = (\mathbf{s}_0, \ldots, \mathbf{s}_n)$ from $\mathcal{G}_c$
  if $\mathbf{s}_n \in \mathcal{S}_{tr}$ **then**
    Add τ to $\mathcal{G}$
  else
    Compute valid actions $\mathcal{A}(\mathbf{s}_n)$
    for each $a \in \mathcal{A}(\mathbf{s}_n)$ **do**
      $\mathbf{s}_{n+1} \leftarrow T(\mathbf{s}_n, a)$
      if $d(\mathbf{s}_{n+1}) = d(\mathbf{s}_n) + 1$ **then**
        $\tau' \leftarrow (\mathbf{s}_0, \ldots, \mathbf{s}_n, \mathbf{s}_{n+1})$
        Add τ' to $\mathcal{G}_c$
      end if
    end for
  end if
end while
return $\mathcal{G}$



(a) Deep Sea Treasure (abstraction)

(b) MineCart



(c) MineTrain

Figure 5: Frames of MORL environments.

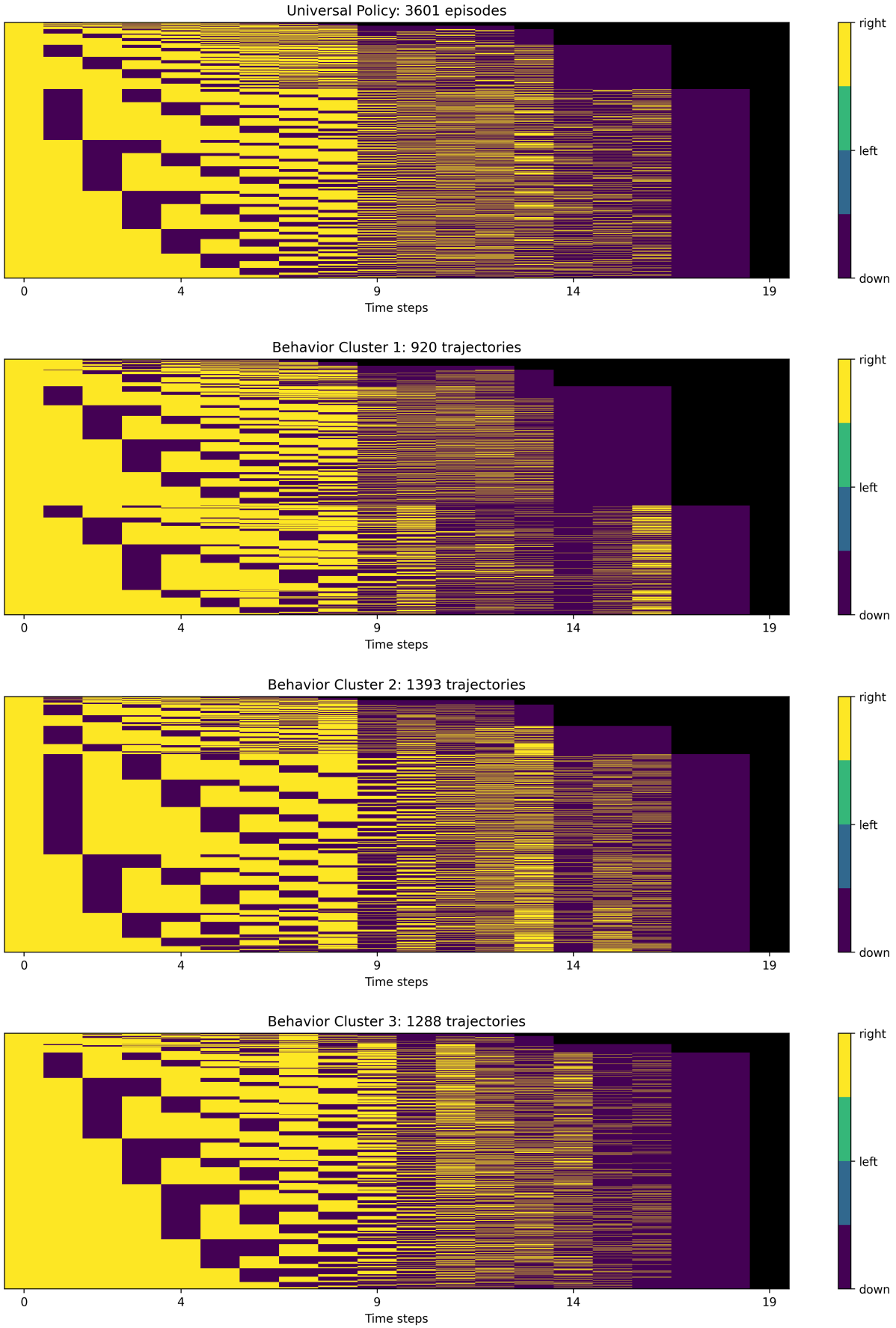


Figure 6: Temporal alignment plot, all the trajectories in Deep Sea Treasure's ground truth (top) in comparison with final clusters, using k-medoids and the agreement features.