The Need for MORE: Need Systems as Non-Linear Multi-Objective Reinforcement Learning

Matthias Rolf
Oxford Brookes University, Oxford, UK
mrolf@brookes.ac.uk

***Abstract*—Both biological and artificial agents need to coordinate their behavior to suit various needs at the same time. Reconciling conflicts of different needs and contradictory interests such as self-preservation and curiosity is the central difficulty arising in the design and modelling of need and value systems. Current models of multi-objective reinforcement learning do either not provide satisfactory power to describe such conflicts, or lack the power to actually resolve them. This paper aims to promote a clear understanding of these limitations, and to overcome them with a theory-driven approach rather than ad hoc solutions. The first contribution of this paper is the development of an example that demonstrates previous approaches' limitations concisely. The second contribution is a new, non-linear objective function design, MORE, that addresses these and leads to a practical algorithm. Experiments show that standard RL methods fail to grasp the nature of the problem and ad-hoc solutions struggle to describe consistent preferences. MORE consistently learns a highly satisfactory solution that balances contradictory needs based on a consistent notion of optimality.**

***Index Terms*—Need systems, multiple objectives, reinforcement learning, value systems**

I. Introduction

Artificial agents are often designed with a plethora of partially contradicting motives, needs, values, and motivations. Some of these are modeled in analogy to biological agents. Self preservation, for instance, translates into models of fear [1], [2], and more particular technical needs like obstacle avoidance [3]. It also relates to self sustenance which translates into consumption of external supplies such as food, water, or battery charge. Cognitive needs include various kinds of curiosity [4], as well as the need for partial control of the environment [5], [6]. Social needs relate to the interaction with other agents, and can be directed as positive emotional interactions [7], empathic interaction with bonded individuals or caretakers [8], [9], or having interactions at all [10], [11]. Other needs of artificial agents or robots do not draw any direct motivation from biology or psychology and rather aim at engineered solutions to practical problems. Most technical systems have at least moderately well defined external tasks such as picking and sorting objects, answering external questions [12], or driving to a particular location on command [13].

Some technical systems are designed to solve exactly one of these needs for a given context. An arsenal of effective methods from planning to optimization and reinforcement learning is available for such cases. However, even engineered systems rarely ever have only one concern, in particular when learning and adaptability is of relevance. If more than one need is relevant to an agent, these needs will eventually produce situations in which interests conflict. A classic example in the design of learning agents is the problem of exploration vs. exploitation [14], or intrinsic vs. extrinsic motivation [4]. A simple but typically inefficient solution is to separate these needs into distinct stages, and let an agent follow intrinsic needs first (exploration), then extrinsic needs. This model was advocated [15] based on staged models of human development [16], but has since then been overcome by more plausible and efficient methods that intertwine these needs [17], [18].

A less abstract and more immediate example of need conflicts is obstacle avoidance in manipulator control [19] or driving in physical [3] or social [13] spaces. Reaching a desired goal state and avoiding obstacles cannot be separated into stages. Both need to be achieved at the same time, but may contradict. Reaching a goal may in fact require close passage by obstacles, or large detours, or may be impossible without collision in certain situations. Conflicts between needs generally get delicate when either self-preservation or ethics are involved [11], [20]. Any robot that moves and locomotes to achieve its tasks is exposed to risk of damage to itself, running out of energy, and may in turn damage the environment or injure bystanders. Yet, these risks can only be entirely avoided if the task is not attempted at all. Even non-physical entities like chat bots may be driven to seriously inappropriate action [21]. Yet, not saying anything would not allow the bot to achieve its task.

A. Rational Agents

The fundamental question of this paper is how such conflicts can be modelled and resolved. The central perspective onto this question is that of rational agents: the eventual decision of an agent should be consistent with a globally defined, scalar *preference* or *utility function* that models which options of behavior are preferable. Important specific contributions to the field have been made also outside of this formulation (notably various cognitive architectures [22] as well as simulated hormonal systems [23]), but lack generalizability in both engineered design as well as analysis of systems. The rational perspective then bears two questions:

1) How can the desired behavior of an agent be formulated effectively by means of a utility function when facing competing needs?

2) How can that utility function be used to effectively and efficiently guide behavior towards those needs?

Standard reinforcement learning addresses the optimization of a single objective over time. A natural first extension to multiple objectives is to simply add up the reward signals corresponding to each objective. If a state has a “task” reward $r_0 = 0.1$ and a “obstacle avoidance” reward $r_1 = -0.3$, then the combined reward presented to the reinforcement learning algorithm is $r = r_0 + r_1 = -0.2$. This idea is easily generalized to weighted contributions such as $r = w_0 \cdot r_0 + w_1 \cdot r_1$ or in vectorial notion $r = \vec{w}\vec{r}$. This scheme is extremely common in the practice of reinforcement learning. Choosing these weights is part of the wider problem of *reward shaping*. The key problem is that the behavior of the agent changes in not always intuitive ways when reward contributions are modified. This is a notorious problem for example in obstacle avoidance, where too close proximity to obstacles is often worth penalizing. However, it is often entirely unclear how large the penalty can be before the robot will simply get stuck on obstacles that would normally be avoidable. From the agents’ perspective this is called *reward hacking* [24]: the agent will optimize what the objective function says, not what the designer intended.

The common approach is to use linear combinations of reward signals both across objectives (by weighting) and across time, which is modeled by future reward discounting in the Bellman equation. Non-linear approaches have been studied, though. Non-linear approaches for single objectives include multiplicative rewards [25], or distributional reinforcement learning [26]. More general non-linear utility or value functions to describe the value of any state in a Markov decision process can help to model the designer’s intention. However, non-linear multi-objective utility functions generally require at least modifications to the Bellman equation on which modern reinforcement learning rests, and typically lead to non-stationary solutions [27] which are extremely difficult to capture with a general purpose algorithm (see [28] for a survey). An alternative approach is to avoid any reduction, or ”scalarization”, to a single utility. Multiple-policy multi-objective reinforcement learning [28] instead aims to find policies for different or all possible compromises between objectives. The most general approach searches for the entire front of *Pareto-optimal* policies [29], [30]. A common alternative is to search for the linear convex hull of all possible compromises [31], which is a subset of all Pareto-optimal policies. Both approaches require a clearly defined space of candidate policies, which means they do not automatically discover non-stationary solutions emerging from non-linear utility functions. More importantly, multiple policies provide no actual resolution of conflict, but only describe feasible alternatives and defer the actual decision about them.

A separate alternative are models with dynamically changing linear weights [32], [33]. Where future weight changes are predictable, they can be incorporated into the planning process [34]. However, how to design sensible weight dynamics is far from obvious. Non-linear processes such as saturation of needs [35] have been demonstrated to be important for balancing. Studies so far have used heuristically designed or externally provided weight or priority dynamics. No study so far has shown that those dynamics are consistent with any general notion of preference or optimality.

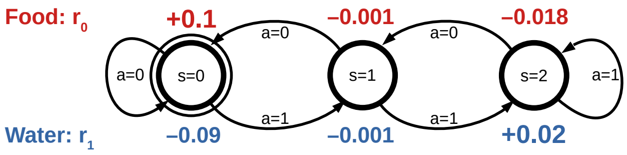

Fig. 1. Example problem: a multi-objective MDP with three states and two actions. The initial state is $s = 0$, where a gain of the first objective (red) can be achieved at the cost of the second objective (blue). In reverse, gains on the second objective can be achieved in $s = 2$, but which requires to move through $s = 1$, in which both objectives lose slightly.

B. Approach and Outline

This paper adopts the rational approach to need systems and starts from a multi-objective reward notation. There is not a single, universal way in which conflicting objectives need to be resolved. The central design assumption of this paper is that needs and their objective values require a *balanced* resolution, meaning that all needs need to be fulfilled to an extent. No need must be fully neglected in order to optimize another. If faced with both an external task and obstacle avoidance, the agent needs to achieve *both*, not simply an arbitrarily weighted average utility between both. If a biological being needs food and water to survive, then it actually needs *both*. It cannot survive based on ever more water at the expense of no food. A chatbot needs to engage with its audience *and* behave ethically acceptable. The design assumption here means that no objective can be arbitrarily traded off for another.

The first contribution of this paper in Sec. II will be an example with which the difficulty and subtlety of this requirement can be concisely studied. A comprehensive analysis of deterministic, stochastic, and also non-stationary solutions to this example will demonstrate that linear scalarizations can fundamentally not address the requirement. The second contribution is a non-linear utility function (”MORE”) that is designed to balance objective contributions, and from which a practical algorithm is directly derived in Sec. III. In particular, it is shown that optimizing MORE corresponds to a very natural weight dynamics which can be analytically derived. Experiments in Sec. IV show that optimizing MORE leads to very satisfactory and non-trivial solutions on the example problem. The paper will conclude with a discussion of some interesting connections between previous approaches that are revealed by the derivation of this algorithm, and discuss questions emerging from it in regards to further theorization and unification as well as practical application.

II. Opposing needs in three states

The example problem proposed for study is shown in Fig. 1. The underlying state and action structure is a deterministic Markov process with three states and two actions. Two objectives are modelled by attributing a scalar reward signal for every combination of state and objective. The first objective can only be accumulated in the first state, while loss is incurred on the second objective. The loss of the second objective is slightly smaller than the gain of the first, so that an overall gain (positive sum) can be achieved. In reverse, a gain of the second objective can only be achieved on the end ($s=2$), but with a loss on the first objective. The loss in both cases is calibrated as -0.9 times the gain of the other. The objective values in $s = 2$ are overall five times lower than in $s = 0$, which means that achieving any kind of balance will require asymmetric treatment of both states. In between, the agent has to pass through a state in which a small loss is incurred on both objectives. A satisfactory solution in design terms would need to make the agent alternate between $s=0$ and $s=2$ in some way. Any solution that stays on one of the sides will indefinitely lose on one objective, which is against our overall point of balanced need achievement that does not fully neglect any one need.

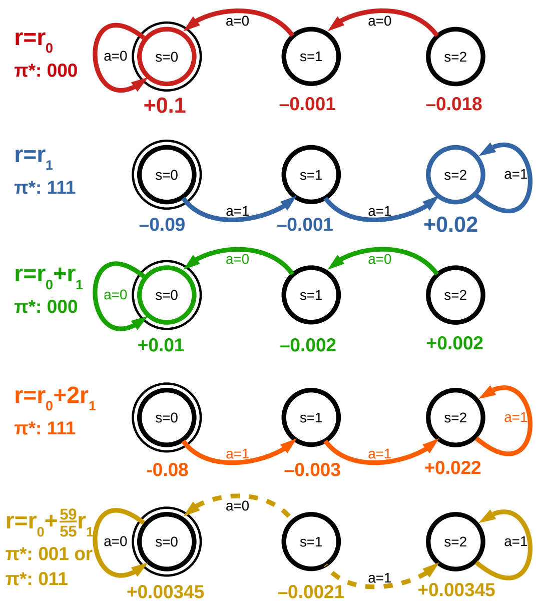

Fig. 2. Various linear scalarizations (objective weightings) are shown together with their optimal deterministic policies on the given state space. All cases lead to the agent moving to either $s=0$ or $s=2$ and then staying there. At the middle point both states have equal value (last row), which requires a tie break in $s=1$ before the agent gets stuck. The policy codes, e.g. ”001” refer to which action is chosen in which state, and match the labels in Fig. 3.

This example fundamentally models two aspects: first, it models conflicting needs. It contains no “easy” solution or state in which the agent could rest while achieving both objectives. Second, the middle state models that transitions between objectives are not free of cost, which will turn out to be a critical aspect for developing a clear understanding of optimality. Planning, as well as the right space of candidate policies are strictly necessary, as the first step of analysis will show that deterministic policies on the given state space are all unsatisfactory. This example is deliberately not modelled as “realistic” depiction of any practical situation. Rather, it is meant to distill key difficulties of practical situation into a somewhat exaggerated value conflict, all while fitting into a small size MDP. Both the quantitative exaggeration and the small size serve the purpose of this study: to comprehensively analyze the nature of value conflicts and the implications of different notions of optimality.

For illustrative purposes Fig. 1 mentions ”Food” and ”Water” as possible real-world quantities to be collected by an agent, which may help to attribute concrete meaning. This paper in no way relies on this interpretation, though. Other values that could correspond to these objectives are some external task as r_0 and battery charge as r_1. In either case, both objectives would have to be maintained, while not being achievable at the exact same time. A more socially and anecdotally stimulating interpretation could be a party gathering in three rooms. The snacks are provided in one room, which makes people thirsty. To get drinks (which make them hungry) they have to go to the other room, after having to pass through a corridor in which neither is provided.

A. Linear Utility Solutions

With three states and two actions, this problem has exactly $3^2 = 8$ different deterministic policies π. They will be denoted by codes such as ”101”, indicating the action choices $\pi(s=0)=1, \pi(s=1)=0, \pi(s=2)=1$, which would take the agent from $s = 0$ to $s = 1$ by choosing $\pi(s = 0) = 1$, and then back to $s=0$. From the perspective of the initial state $s=0$, the policies ”000”, ”001”, ”010”, and ”011”, are equivalent because the agent never leaves $s = 0$. The expected future cumulative reward V_k^{LIN} per objective k can be calculated analytically based on the notion of a policy dependent state transition matrix T_{ij}, indicating the probability to go to $s=j$ from $s=i$, and a discount factor γ:

$$\vec{V}_k^{LIN} = E\left[\sum_t \gamma^t r_k(t)\right]_{s(0)=i} = (\mathbb{1} - \gamma \cdot T)^{-1} \cdot \vec{R}_k \ , \quad (1)$$

where $\vec{R}_k$ is the vector listing the rewards of objective k for all states, $\vec{V}_k^{LIN}$ contains the values for all states, and $\mathbb{1}$ is the identity matrix [14].

The optimal deterministic policies for various linearizations $V^{LIN} = w_0 {\cdot} V_0^{LIN} + w_1 {\cdot} V_1^{LIN}$ are shown in Fig. 2. For equal weights $w_0 = w_1 = 1$, the first state will be most individually valuable because that state has the most positive margin. The exact value depends on the choice of γ, but for all moderately high values ($\gamma = 0.9$ is used for all examples) the optimal policy will be ”000” leading straight into that state. Higher weight may be given to r_1, which leads to $s=2$ taking over with ”111” as optimal policy. The last row of Fig. 2 shows the tipping point where left and right state have the same momentary payoff, and the agent will stay on which ever side it starts. The exact values of all policies are plotted in Fig. 3. The eight deterministic policies are all Pareto optimal on this problem: no policy is better in *both* objectives than any other. All eight deterministic policies are unsatisfactory, though, as none of them allows to cycle between $s=0$ or $s=2$ to achieve balance.

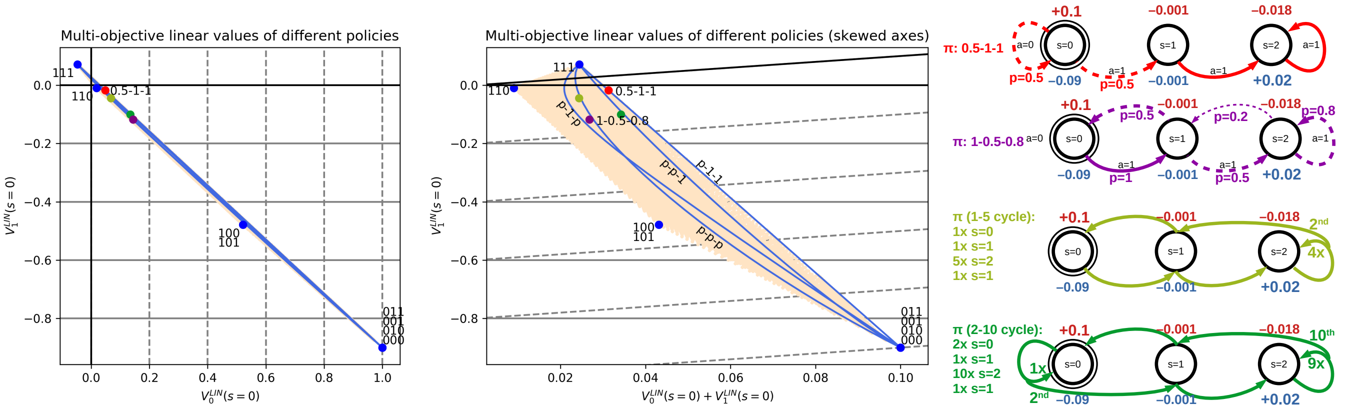

Fig. 3. Value plot of a range of policies. The left side shows a regular plot of the individual values. The center plot shows the same data on different axes that decompress the narrow structure on which all policies lie. Dashed and solid grey grid lines in both plots exactly correspond to each other. Blue dots represent all deterministic policies. The beige area is the space of all stochastic policies. Particular stochastic policies generated by mixing policies on the corners are shown as blue lines. Red, purple, and green dots correspond to stochastic and non-stationary policies displayed on the right.

Stochastic policies can help to improve the situation. Stochastic policies in our example take the shape "p_0-p_1-p_2", indicating the probability p_0 to choose action $a=1$ in state 0, p_1 in state 1, and p_2 in state 2. In this notation the stochastic policy "1-1-1" is equivalent to the deterministic policy "111". Fig. 3 shows the performance of the entire space of stochastic policies in beige. Data was gathered by sampling each of the three probabilities with 30 intermediate steps and applying the same analytic evaluation as above. As is well known in multi-objective optimization, the resulting shape is the convex hull of the performance of the deterministic policies [28]. This can be derived by regarding each stochastic policy as a convex mixture of several deterministic policies.

The Pareto front of that space is the upper right flank of the convex hull. Data shows that this side is precisely formed by policies of the shape "p-1-1", which interpolate between "011" and "111". It would initially seem that also mixtures of other lower-right vertex policies such as "000" with "111" could lie on that edge, but these create new possible loops in the transition graph that result in higher-order, spline-like interpolations. In this example, they do not lie on the Pareto front. The linearly optimal "p-1-1" policies (see red example in Fig. 3) can create better balance between the objectives by staying in $s=0$ for some time, and then transitioning to $s=2$, but the agent will again be stuck in $s=2$. They still fundamentally fail to maintain balance between the objectives. Interestingly, there *are* stochastic policies that at least in a weak stochastic sense balance the objectives in the long run. An example is shown in the second row of Fig. 3 (right) in purple. These policies let the agent make a balanced random walk through the state graph, which will ultimately reach both objectives arbitrarily often. The policies are, however, not linearly optimal under any weighting. In the graph (Fig. 3) they appear distant from the Pareto front and convex hull in the middle of the space. The reason for this sub-optimality under linear notion is that they traverse the center state $s=1$ more often than the others, in which both objectives lose.

B. Discussion

All linearly optimal solutions to this problem fail to achieve the initial design goal: balancing the needs. The analysis of stochastic policies has shown that this is not because of the degeneracy of the state graph, which forces a left/right decision of the agent in the middle. Policies that choose mixtures here simply do not come out optimal. The general reason is that *linear optimization discourages balance*: contributions from either objective are treated equal as long as the margin is large enough. Balance is immediately ruled out as soon as any cost of transition between the objectives is required.

Besides random-walk-like stochastic policies, this problem invites certain kinds of non-stationary policies. A deterministic cyclic walk with the structure of staying in $s=0$ for L steps, then transitioning and staying in $s=2$ for 5L steps and going back to $s=0$ seems an attractive solution. Fig. 3 (right) shows examples for $L=1$ (3rd row) and $L=2$ (4th row). It is easy to calculate that these walks balance the objectives in the long run and even achieve positive gain: a 1-5 cycle stays once in 0, twice in 1, and five times in 2 per cycle. The cumulative reward for each closed cycle is $0.1 - 0.001 \cdot 2 - 0.018 \cdot 5 = -0.09 - 0.001 \cdot 2 + 0.02 \cdot 5 = +0.008$ for *both* objectives. Yet, it does not come out as linearly optimal (see Fig. 3) due to the ongoing transition cost.

This example shows clearly that linear scalarizations cannot sufficiently model *or* resolve conflicting needs. A non-linear scalarization is necessary, which may dictate non-stationary solutions such as the examples described in the previous paragraph. The next section will demonstrate the non-linear MORE scalarization that leads to an augmented state space which can indeed describe such policies.

III. Multi-Objective Reward Exponentials (MORE)

Instead of considering a weighted sum of objective values, this paper suggests to use an exponential value expression per each objective. This section will first demonstrate how this leads to the idea of "softmin" like value function that

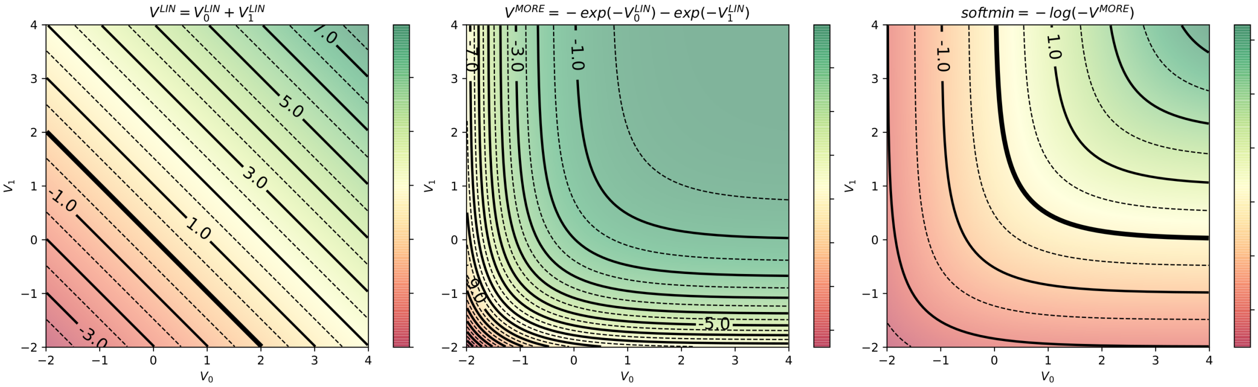

Fig. 4. The geometry of different utility functions for two objectives. The linear utility (left) treats contributions from all objectives equally. Equivalent choices therefore have the shape of straight lines. MORE (center) has diminishing returns in each objective. It is monotonically increasing in each objective, but the slope is dominated by the least achieved objective. The logarithmic transform of MORE (right) models a soft-min function. High overall values are only achieved if *both* individual values are high.

provides an elegant view on the need balancing problem. Then, an augmented state variable is derived that allows to express policies which are non-stationary with respect to the original Markov process. Finally, an algorithm is developed which solves the optimization by adding only a thin non-linear layer on top of ordinary Q-learning.

Developing a consistent utility function first requires to model the value of a single roll-out of a policy, such as a list of states, actions, and rewards over time $L = (s_t, a_t, r_t)_{t=0\dots T-1}$. The proposed equation for the multiple objective reward exponentials (M.O.R.E.) function on a roll-out is:

$$V^{MORE}(L) = -\sum_{k=0}^{K-1} \exp\left(-V_k^{LIN}(L)\right) \quad (2)$$

$$= -\sum_{k=0}^{K-1} \exp\left(-\sum_t \gamma^t r_k(t)\right) \quad . \quad (3)$$

The only addition to the traditional linear scalarization are the exponential functions, with a negation on both sides. Terms of the form $-exp(-V)$ are in fact used commonly in single objective optimization as a model of risk aversion [36], but seem to have never been applied to a multi-objective setting. The term $-exp(-V)$ models rapidly diminishing returns: very high values of V quickly saturate $-exp(-V)$ close to zero. This is similar to previous ad-hoc models of need saturation [23], [35]. Large negative V, however, leads to large negative $-exp(-V)$ which can be seen as deficit model. The expression therefore gains a new significance in the multi-objective setting: objectives with large deficits will dominate V^{MORE} and incentivize balancing. All other objectives still count towards the total, but with less influence.

Applying the logarithm (which due to monotonicity does not change preference) to V^{MORE} does in fact result in a *soft minimum* like function, see Fig. 4. When the objectives have largely different values, MORE is primarily driven by the minimum, and $-log(-V^{MORE})$ approaches the minimum single objective value in the limit because other exponential terms vanish. This is similar to a hard-minimum approach [37], but which entirely neglects all terms but the minimum. MORE, in contrast, continuously weights in all objectives, and resembles the linear objective for nearly balanced situations.

A. State and Action Values

A well defined roll-out value function does not automatically result in a value function that is well defined on states of the underlying Markov process. Linear value functions can be completely expressed by only knowing the MDP's state s. Bellman's seminal work showed that future and past value are completely separable, and that no other information has to be known about the past other than that it led to s in order to make an optimal decision about the future. Knowing only the current MDP state s is *not* sufficient to determine the optimal future policy under MORE. The past may have accumulated a deficit in r_0, which requires to focus on optimizing r_0 in the future, but which the state s cannot tell. However, a natural state variable exists that exactly models and tracks such deficits. We can derive it by considering a roll-out $L = P{+}F$ consisting of a "past" P and a subsequent "future" part F. Inserting those into the MORE equation above gives:

$$-\sum_{k=0}^{K-1} \exp\left(-\sum_{t=0}^{T_P-1} \gamma^t r_k^P(t) - \sum_{t=0}^{T_F-1} \gamma^{T_F+t} r_k^F(t)\right)$$

The inner sum can be pulled into a product of exponentials that exactly separate past and future:

$$-\sum_{k=0}^{K-1} \exp\left(-\sum_{t=0}^{T_P-1} \gamma^t r_k^P(t)\right) \cdot \exp\left(-\gamma^{T_P} \sum_{t=0}^{T_F-1} \gamma^t r_k^F(t)\right)$$

The left exponential is the state variable which naturally accumulates all necessary of the past, and has a very natural *interpretation as dynamic weight*:

$$V^{MORE}(P{+}F) = -\sum_{k=0}^{K-1} w_k(P) \cdot \exp\left(-\gamma^{T_P} V_k^{LIN}(F)\right) \quad (4)$$

In vectorial notion we can write:

$$V^{MORE}(P{+}F) = -\vec{w}(P)\cdot\exp\left(-\gamma^{T_P}\vec{V}^{LIN}(F)\right) \quad . \tag{5}$$

Thus, we can define a state value function based on an *augmented MDP* which instead of only s has $(s, \vec{w})$ as state. The transition between $s(t)$ and $s(t+1)$ remains as specified by the original MDP and is controlled by the actions $a(t)$. The weights are determined to track the running exponential deficit of each objective, which is not determined by the action $a(t)$ or state $s(t)$ as such, but by the previous weight and the current reward:

$$\vec{w}_k(t+1) = \vec{w}_k(t)^{\gamma} \cdot \exp(-r_k(t)) \tag{6}$$

With this augmented state it is straightforward to define a state value function for any given policy π:

$$V_{\pi}^{MORE}(s, \vec{w}) = -\vec{w}\cdot\exp\left(-\vec{V}_{\pi}^{LIN}(s, \vec{w})\right) \tag{7}$$

The discounting is encapsulated inside the linear value and the weight dynamics in this case. Remarkably, the standard linear value estimation for each separate objective remains intact inside this equation. This means that standard, linear value estimation methods such as SARSA or Q-learning can be used to estimate the future returns of the individual objectives. Only a thin non-linear layer has to be wrapped around it to estimate the MORE value. This requires the addition for continuous state variables $\vec{w}$, though. This means that purely tabular, and therefore "exact" optimization is not possible anymore. For problems that are already continuous or require function approximation to estimate rewards, this only means the addition of few extra variables, one per objective.

B. MORE-Q Learning

A natural analog of Q learning can be formulated based on the above value equation. First we define a state-action value function mirroring the state-value function above:

$$Q^{MORE}(s, \vec{w}, a) = -\vec{w}\cdot\exp\left(-\vec{Q}^{LIN}(s, \vec{w}, a)\right) \tag{8}$$

A function approximator needs to be used to model $\vec{Q}^{LIN}$, with inputs $(s, \vec{w}, a)$ and K-dimensional output representing the value of each separate objective.

We assume a sample $((s, \vec{w}), a, \vec{r}, (s', \vec{w}'))$ has been generated off-policy. $(s, \vec{w})$ represents the state in which action a was chosen. $\vec{r}$ is the payoff vector containing the immediate pay-offs of each objective. $(s', \vec{w}')$ is the next state, where s' is generated by the world, and $\vec{w}'$ updated according to Eqn. 6. As with ordinary Q-learning, the first step is to determine the optimal action for $(s', \vec{w}')$:

$$a^* = \underset{a}{\operatorname{argmax}}\ Q^{MORE}(s', \vec{w}', a) \tag{9}$$

Then, an ordinary linear value update is performed based on that action choice

$$\vec{Q}^{LIN}(s,\vec{w},a) \leftarrow (1{-}\alpha)\vec{Q}^{LIN}(s,\vec{w},a) + \alpha\Big(\vec{r} + \gamma\cdot\vec{Q}^{LIN}(s',\vec{w}',a^*)\Big)$$

with some learning rate α.

IV. Results

Experimental results on the example problem (Fig. 1) compare three different algorithms. The first baseline is standard linear Q-learning, for which the rewards r_0 and r_1 are simply added. We already know from Sec. II that the outcome of that should be the policy "000", but it is useful as a baseline nevertheless. The second algorithm is a simple objective switching mechanism based on separate Q-learning for each objective, similar to previous "winner-takes-all" architectures [38]. It Q-learns one policy π_0 to optimize only r_0, and one policy π_1 to optimize only r_1. Whenever the past cumulative reward is lower for the first objective, it picks an action from π_0. If the cumulative reward is lower for the second objective it picks π_1. Hence, this strategy greedily aims for the objective that is currently least achieved.

The third algorithm is the MORE-Q algorithm described above. Standard locally linear regression is used to model the linear Q values $\vec{Q}^{LIN}(s, \vec{w}, a)$. Because of the small tabular structure of this problem, all six state-action pairs are given their own local linear map with only $\vec{w}$ as input. The distance between local models is set to a Euclidean distance of 0.4 in weight space. In order to achieve bounded inputs that can be effectively used for function approximation, the weights are normalized to $\sum_k w_k = 1$ in every step, which can be shown to not change the overall dynamics or preferences.

All three algorithms are based on the same underlying Q learning implementation. Updates are consistently performed with a learning rate $\alpha = 0.2$. Exploration and learning are performed for 20.000 steps of epsilon-greedy exploration. The chance of a random action is 50% at first, and then halves every 2.000 steps. The discount of future rewards is set to $\gamma = 0.9$ for all algorithms. In addition to training on the current sample, training is performed on ten randomly selected past samples in every step. The objective switching baseline as well as MORE-Q require the calculation of accumulated past reward. Because accumulating past reward over long time frames is easier than predicting over long time frames, the "backwards" oriented discount factor used to update $\vec{w}$ (Eqn. 6) is set to a larger value of $\gamma^{past} = 0.99$. This ensures that needs can be balanced over longer time windows.

Fig. 5 shows the results based on three statistics, each shown for ten independent trials for which the curves are overlaid. The first statistic shows how many consecutive time-steps the agent stays in each of the three states once (and if) is enters it. Linear Q-learning stays ever longer in the first state, only occasionally interrupted when epsilon-greedy exploration forces it out. The average run length of the other two states is one, meaning those states are left immediately even if explored. This results in a continuous accumulation of the first objective at the expense of the second, as shown by the exponential running average across 1.000 steps of both objectives in the second plot. The "softmin" metric is strongly negative in this case.

The objective switching method (second row of plots) balances the objectives very exactly in this case. As shown

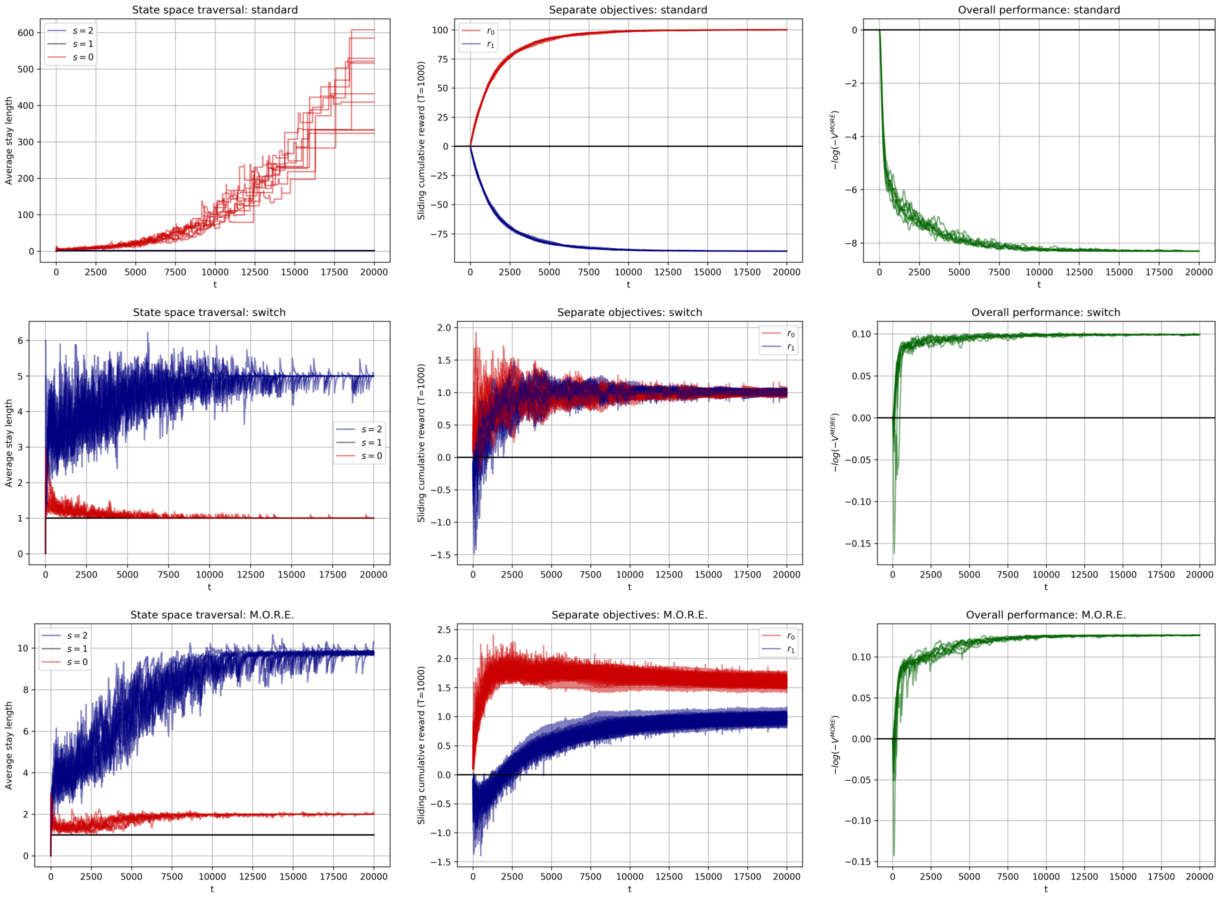

Fig. 5. Results for standard linear Q learning (top row), objective switching (middle row), and MORE-Q learning (bottom row). The first column shows the average stay length in each state, which reveals the strategy taken by each approach. The second and third columns for the resulting performance for the individual objectives as well as the overall MORE utility.

earlier, the optimal policies for the separate objectives are ”000” and ”111”. This method learns them separately and switches between them. Because $s=2$ has a five times lower margin than $s = 0$, this method spends exactly five times more time there. After staying in 0 one single step the method detects a deficit in r_1 and moves to $s=2$. There it takes five steps to exactly nullify that deficit, and the method switches back to maximizing r_0 in $s=0$. Hence, this method results in the 1-5 cycle already analyzed in Sec. II. This solution is in fact quite suitable for this particular problem. However, this method represents a purely greedy strategy only looking for one objective at a time. The switching method is therefore very easily fooled by any kind of local optima and cannot estimate policies that truly optimize combinations of objectives.

MORE-Q is able to model and estimate such policies and actually discovers the 2-10 cycle: it stays for 10 steps once it reaches $s=2$ and for two steps in $s=0$. This is arguably a slightly better strategy in terms of overall design intentions, and pays off with a slightly higher empirical V^{MORE} score (right plot). This pays off because less time is spent in the middle state. Longer cycle lengths such as 3-15 do not pay off further within the planning horizon implied by the discount factor of $\gamma = 0.9$ (corresponding to $\frac{1}{1-\gamma} = 10$ time steps) used in these experiments. Further simulation runs (not plotted) show, however, that MORE-Q discovers even longer cycles such as 6-30 for $\gamma = 0.99$.

V. Conclusion

Linear scalarizations are not suitable to satisfy multiple competing needs in a balanced way. This paper has introduced a novel, non-linear utility function: MORE. It has been shown to resemble a softmin function which incentivizes to focus on improving on the least achieved objective without fully neglecting the other needs. If all rewards are very small (close to zero), this function contains the standard linear scalarization as limit case. For very large values it approaches a hard-minimum function. In a reinforcement learning setting this has been shown to lead to very naturally changing weights that prioritize the different objectives. While previous approaches have used ad-hoc designs for such weight changes or required externally specified weights, MORE allows to derive optimal weights coherently based on first principles.

This paper has deliberately not addressed large problems, but aimed at fully working out a small but difficult one and understanding key difficulties. The example developed and analyzed here has only three states, but has remarkable subtlety. Linear approaches fail entirely in capturing its challenge, whereas the non-linear MORE allows to describe and learn a very satisfying solution. The question of scalability obviously has to be asked in future work. Since the MORE-Q learning algorithm derived from the MORE utility heavily relies on standard Q-learning with only some non-linear addition, MORE-Q could be generally as scalable as Q-learning. The only true complexity addition is the augmentation of the state space with weights for each objective, but for which the analytic dynamics are perfectly known.

Apart from such practical questions, a potentially interesting future line of investigation is further theorization and unification with existing approaches, including decision problems in economics [39] and neuroscience [40]. For example, can MORE be related to previous models of fear and self-preservation [1], [2] that showed benefits from learning positive and negative stimuli separately? Can previously handcrafted need systems based on hormonal [23] or motivation systems [35], or affective interaction [7] be generalized or theorized with the aid of MORE? Future work should investigate how even "MORE" can be gained from approaching developmental problems based on multi-objective learning.

References

[1] N. Navarro-Guerrero, R. J. Lowe, and S. Wermter, "The effects on adaptive behaviour of negatively valenced signals in reinforcement learning," in *IEEE ICDL-EpiRob*, 2017.

[2] B. Seymour and S. Elfwing, "Parallel reward and punishment control in humans and robots: Safe reinforcement learning using the maxpain algorithm." in *IEEE ICDL-EpiRob*, 2017.

[3] J. Borenstein, Y. Koren *et al.*, "The vector field histogram-fast obstacle avoidance for mobile robots," *IEEE transactions on robotics and automation*, vol. 7, no. 3, pp. 278–288, 1991.

[4] P.-Y. Oudeyer, F. Kaplan, and V. V. Hafner, "Intrinsic motivation systems for autonomous mental development," *IEEE transactions on evolutionary computation*, vol. 11, no. 2, pp. 265–286, 2007.

[5] V. Charisi, D. Davison, F. Wijnen, J. Van Der Meij, D. Reidsma, T. Prescott, W. Van Joolingen, and V. Evers, "Towards a child-robot symbiotic co-development: a theoretical approach," in *AISB Convention 2015*, 2015.

[6] K. Fischer, K. Lohan, J. Saunders, C. Nehaniv, B. Wrede, and K. Rohlfing, "The impact of the contingency of robot feedback on hri," in *2013 International Conference on Collaboration Technologies and Systems (CTS)*. IEEE, 2013, pp. 210–217.

[7] C. Breazeal, "Affective interaction between humans and robots," in *European Conference on Artificial Life*. Springer, 2001, pp. 582–591.

[8] M. Asada, "Towards artificial empathy," *International Journal of Social Robotics*, vol. 7, no. 1, pp. 19–33, 2015.

[9] J. L. Copete, Y. Nagai, and M. Asada, "Motor development facilitates the prediction of others' actions through sensorimotor predictive learning," in *IEEE ICDL-EpiRob*. IEEE, 2016.

[10] L. Süssenbach, N. Riether, S. Schneider, I. Berger, F. Kummert, I. Lütkebohle, and K. Pitsch, "A robot as fitness companion: towards an interactive action-based motivation model," in *IEEE Int. symposium on robot and human interactive communication*, 2014.

[11] M. Rolf, N. Crook, and J. Steil, "From social interaction to ethical ai: A developmental roadmap," in *IEEE ICDL-EpiRob*, 2018.

[12] R. High, "The era of cognitive systems: An inside look at ibm watson and how it works," *IBM Corporation, Redbooks*, pp. 1–16, 2012.

[13] T. Kruse, A. K. Pandey, R. Alami, and A. Kirsch, "Human-aware robot navigation: A survey," *Robotics and Autonomous Systems*, vol. 61, no. 12, pp. 1726–1743, 2013.

[14] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[15] D. Bullock, S. Grossberg, and F. H. Guenther, "A self-organizing neural model of motor equivalent reaching and tool use by a multijoint arm," *Journal of Cognitive Neuroscience*, vol. 5, no. 4, pp. 408–435, 1993.

[16] J. Piaget, *The Origin of Intelligence in the Child*. Routledge and Kegan Paul, 1953.

[17] M. Rolf, J. J. Steil, and M. Gienger, "Goal babbling permits direct learning of inverse kinematics," *IEEE Transactions on Autonomous Mental Development*, vol. 2, no. 3, pp. 216–229, 2010.

[18] A. Baranes and P.-Y. Oudeyer, "Active learning of inverse models with intrinsically motivated goal exploration in robots," *Robotics and Autonomous Systems*, vol. 61, no. 1, pp. 49–73, 2013.

[19] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," in *Autonomous robot vehicles*. Springer, 1986, pp. 396–404.

[20] C. Allen and W. Wallach, *Robot ethics: the ethical and social implications of robotics*. MIT press, 2011.

[21] M. J. Wolf, K. Miller, and F. S. Grodzinsky, "Why we should have seen that coming: comments on microsoft's tay" experiment," and wider implications," *ACM SIGCAS Computers and Society*, vol. 47, no. 3, pp. 54–64, 2017.

[22] R. Brooks, "A robust layered control system for a mobile robot," *IEEE journal on robotics and automation*, vol. 2, no. 1, pp. 14–23, 1986.

[23] J. Lones, M. Lewis, and L. Canamero, "From sensorimotor experiences to cognitive development:: How does experiential diversity influence the development of an epigenetic robot?" *Frontiers in Robotics and AI*, 2016.

[24] D. Amodei, C. Olah, J. Steinhardt, P. Christiano, J. Schulman, and D. Mané, "Concrete problems in ai safety," *arXiv preprint arXiv:1606.06565*, 2016.

[25] U. G. Rothblum, "Multiplicative markov decision chains," *Mathematics of Operations Research*, vol. 9, no. 1, pp. 6–24, 1984.

[26] W. Dabney, M. Rowland, M. G. Bellemare, and R. Munos, "Distributional reinforcement learning with quantile regression," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[27] D. White, "Multi-objective infinite-horizon discounted markov decision processes," *Journal of mathematical analysis and applications*, vol. 89, no. 2, pp. 639–647, 1982.

[28] D. M. Roijers, P. Vamplew, S. Whiteson, and R. Dazeley, "A survey of multi-objective sequential decision-making," *Journal of Artificial Intelligence Research*, vol. 48, pp. 67–113, 2013.

[29] K. Van Moffaert and A. Nowé, "Multi-objective reinforcement learning using sets of pareto dominating policies," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 3483–3512, 2014.

[30] S. Parisi, M. Pirotta, and M. Restelli, "Multi-objective reinforcement learning through continuous pareto manifold approximation," *Journal of Artificial Intelligence Research*, vol. 57, pp. 187–227, 2016.

[31] L. Barrett and S. Narayanan, "Learning all optimal policies with multiple criteria," in *Proceedings of the 25th international conference on Machine learning*, 2008, pp. 41–47.

[32] S. Natarajan and P. Tadepalli, "Dynamic preferences in multi-criteria reinforcement learning," in *Proceedings of the 22nd international conference on Machine learning*, 2005, pp. 601–608.

[33] S. Abdelfattah, K. Kasmarik, and J. Hu, "A robust policy bootstrapping algorithm for multi-objective reinforcement learning in non-stationary environments," *Adaptive Behavior*, p. 1059712319869313, 2019.

[34] A. Abels, D. M. Roijers, T. Lenaerts, A. Nowé, and D. Steckelmacher, "Dynamic weights in multi-objective deep reinforcement learning," *arXiv preprint arXiv:1809.07803*, 2018.

[35] G. Konidaris and A. Barto, "An adaptive robot motivational system," in *International Conference on Simulation of Adaptive Behavior*. Springer, 2006, pp. 346–356.

[36] J. W. Pratt, "Risk aversion in the small and in the large," in *Uncertainty in economics*. Elsevier, 1978, pp. 59–79.

[37] P. Vamplew, R. Dazeley, C. Foale, S. Firmin, and J. Mummery, "Human-aligned artificial intelligence is a multiobjective problem," *Ethics and Information Technology*, vol. 20, no. 1, pp. 27–40, 2018.

[38] B. Girard, V. Cuzin, A. Guillot, K. N. Gurney, and T. J. Prescott, "A basal ganglia inspired model of action selection evaluated in a robotic survival task," *Journal of integrative neuroscience*, vol. 2, no. 02, pp. 179–200, 2003.

[39] A. Tversky and E. Shafir, "Choice under conflict: The dynamics of deferred decision," *Psychological science*, vol. 3, no. 6, 1992.

[40] S. M. Kim and L. M. Frank, "Hippocampal lesions impair rapid learning of a continuous spatial alternation task," *PLoS One*, vol. 4, no. 5, 2009.